1990年 6 月 1 日発行（毎月 1 回 1 日発行）第 9 巻 6 号通巻98号　昭和58年11月 2 日第三種郵便物認可

PERSONAL COMPUTER MAGAZINE for MZ, X1, and X68000

5"2HD標準装備 創刊8周年記念PRO-68K

PurePASCAL/GNU C/ソースジェネレータDIS.X

X68000&PC-286対応S-OS"SWORD"/3D CGシステムANGEL

MUSICDRV.X/OPMD.X/ブロックゲームYet Another Column

X1turbo用コマンドシェルシミュレータINTEGRAL X1

6

1990

SOFT BANK オー／エックス
特別定価780円（付録：5"2HD）

X68000

SX-WINDOW

ひらかれた知性。

見はてぬ夢の象徴。

次代のインテリジェンス、"SX-WINDOW"※搭載。

※SX-WINDOWの起動には、メインメモリ2MBが必要です。PROIIシリーズ(CZ-653C/663C)でSX-WINDOWをご使用の際には、あらかじめ別売の1MB増設RAMボードCZ-6BE1Bを増設してください。

●いま、こだわり続けてきたある執着がまさに帰結しようとしています。グラフィカルユーザーインターフェイス“SX-WINDOW ver.1.0”。もちろん、X68000には発売当初よりビジュアルシェルが搭載されていたことはご存じのとおりですが、クオリティグラフィックやマルチメディア、マルチタスク対応など真の意味での汎用性を志向した開発コンセプトからは、私たち自身ものたりなさを禁じ得なかったことも事実です。しかし、キャラクタユーザーインターフェイス全盛のその時代に、デスクトップの概念をいち早く採り入れた先見性は、現在のインターフェイスの在り方に対する的確な予測に基づくもので、何よりも、トレンディなユーザーの圧倒的な支持によって証明されています。パーソナルコンピュータがその意味どおり、個人のためのツールなら、インターフェイスの発展は必然です。このウインドウシステムは、私たちX68000開発プロジェクトに携わったすべてのスタッフの指標であり、義務でもあったのです——。

●ユーザー本位の操作環境を提供するフル画面マルチウインドウタイプのデスクトップ(テキスト面/単色4階調＋カラー4色。グラフィック面/カラー65,536色中16色)、新感覚スクロールバー…こだわりの美学で高められたユーザーインターフェイス。イベント・ドリブン型マルチタスク処理により複数の作業を同時に処理できる疑似マルチタスクや入出力装置の設定が簡単におこなえる多機能コントロールパネルを搭載した本格ウインドウシステムです。

●“SX-WINDOW”、このひらかれた知性は、今もそしてすぐ後に続く時代をも包含した質の高い「愉しみ」を提供するインターフェイスです。フレンドリーOS Human68kはここに、当初の目的の成就と共に、将来へ確かな展望を明示したといえるでしょう。さまざまなジャンルへ、拡がりと密度を高めるアプリケーション環境、インテリジェントなペリフェラル環境。こうしたトレンドを背景に、いま第4世代のX68000がデビューします。

NEW X68000
PERSONAL WORKSTATION

SUPER・EXPERT・PRO

ザ・ワークステーション。80Mバイトハードディスク、SCSIインターフェイスを標準装備。
SUPER HD 本体＋キーボード＋マウス・トラックボール
HDタイプ CZ-623C-TN(チタンブラック) 標準価格498,000円(税別)〈6月発売予定〉

アートの系譜。**EXPERT II** 本体＋キーボード＋マウス・トラックボール
CZ-603C-BK(ブラック)・-GY(グレー) 標準価格338,000円(税別)/HDタイプ CZ-613C-BK(ブラック) 標準価格448,000円(税別)

ニュースタンダード。**PRO II** 本体＋キーボード＋マウス
CZ-653C-BK(ブラック)・-GY(グレー) 標準価格285,000円(税別)
HDタイプ CZ-663C-BK(ブラック)・-GY(グレー) 標準価格395,000円(税別)

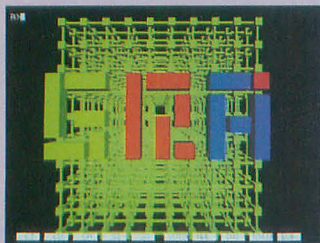
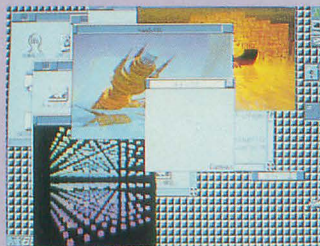
15型カラーディスプレイテレビ(ドットピッチ0.39mm) CZ-602D-BK(ブラック)・-GY(グレー)……………… 標準価格 99,800円(チルトスタンド同梱・税別)
15型カラーディスプレイテレビ(ドットピッチ0.39mm) CZ-605D-BK(ブラック)・-GY(グレー)……………… 標準価格115,000円(スピーカー2個/チルトスタンド同梱・税別)
15型カラーディスプレイテレビ(ドットピッチ0.31mm) CZ-613D-TN(チタンブラック)・-BK(ブラック)・-GY(グレー)…… 標準価格135,000円(スピーカー2個/チルトスタンド同梱・税別)
14型カラーディスプレイ(ドットピッチ0.31mm) CZ-603D-BK(ブラック)・-GY(グレー)……………… 標準価格 84,800円(チルトスタンド同梱・税別)
14型カラーディスプレイ(ドットピッチ0.31mm) CZ-604D-BK(ブラック)・-GY(グレー)……………… 標準価格 94,800円(スピーカー2個/チルトスタンド同梱・税別)

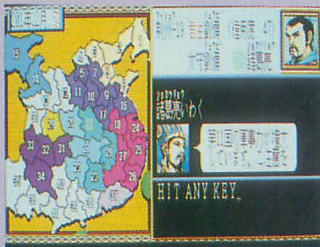
EXEリーダーズグッズ
プレゼント実施中

●いま、EXE会員よりご紹介のお客様がEXEショップでX68000シリーズを購入されますと、EXE会員にEXEリーダーズグッズをプレゼントします。詳しくはEXEショップにお問い合わせください。
●また、X68000シリーズをご購入のお客様は、ぜひEXEクラブにご入会ください。

本広告に掲載しております商品および役務の価格には消費税は含まれておりませんので、ご購入の際、消費税額をお支払い下さい。

●お問い合わせは…シャープ㈱電子機器事業本部システム機器営業部 〒545大阪市阿倍野区長池町22番22号☎(06)621-1221(大代表)
電子機器事業本部液晶映像システム事業部第2商品企画部 〒162東京都新宿区市谷八幡町8番地☎(03)260-1161(大代表)

シャープ株式会社

3D CGAシステムANGEL

グラフィックコンバータSXCONV.X

Yet Another Column

三国志 II

FAR SIDE MOON

シャープ見・体・験フェア

Oh!X

C O N T

●特集

●THE SOFTOUCH

●特別企画

〈スタッフ〉
●編集長／前田 徹 ●編集／植木章夫 太田慎一 岡崎栄子 浅井研二 ●協力／有田隆也 中森 章 後藤貴行 林 一樹 荻窪 圭 岡本浩一郎 毛内俊行 吉田賢司 影山裕昭 相馬英智 古村 聡 村田敏幸 丹 明彦 三沢和彦 長沢淳博 宮島 靖 金子俊一 浦川博之 ●カメラ／杉山和美 ●イラスト／永沢しげる 山田晴久 小栗由香 ●アートディレクター／島村勝頼 ●レイアウト／元木昌子 AD GREEN ●校正／千野延明 織田洋子

表紙絵：塚田哲也

E N T S

1990 JUN.
6

■広告目次

X68000をサポート。

シャープペリフェラルファミリー

X68000

ボード

拡張メモリ

1MB増設RAMボード
(CZ-600C専用)
CZ-6BE1
標準価格 35,000円(税別)

1MB増設RAMボード
(CZ-601C/611C/652C/653C/662C/663C用)
CZ-6BE1B
標準価格 28,000円(税別)

2MB増設RAMボード※4
CZ-6BE2
標準価格 79,800円(税別)

4MB増設RAMボード※4
CZ-6BE4
標準価格 138,000円(税別)

インターフェイス

ユニバーサルI/Oボード
CZ-6BU1
標準価格 39,800円(税別)

GP-IBボード
CZ-6BG1
標準価格 59,800円(税別)

増設用RS-232Cボード
(2チャンネル)
CZ-6BF1
標準価格 49,800円(税別)

数値演算プロセッサ

数値演算プロセッサボード
CZ-6BP1
標準価格 79,800円(税別)

FAX

FAXボード
CZ-6BC1
標準価格 79,800円(税別)

MIDI

MIDIボード
CZ-6BM1
標準価格 26,800円(税別)

ネットワーク

モデム

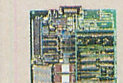
モデムユニット※5
CZ-8TM2
標準価格 49,800円(税別)
(RS-232Cケーブル同梱)

RS-232Cケーブル

RS-232Cケーブル
(平行接続型)
CZ-8LM1
標準価格 7,200円(税別)

RS-232Cケーブル
(クロス接続型)
CZ-8LM2
標準価格 7,200円(税別)

LANボード

NEW

LANボード
CZ-6BL1
標準価格 268,000円(税別)
※電源ユニット・ソフトウェア
(ネットワークドライバVer1.0)同梱

入力

インテリジェントコントローラ
CZ-8NJ2
標準価格 23,800円(税別)

マウス・トラックボール
CZ-8NM3
標準価格 9,800円(税別)

トラックボール
CZ-8NT1
標準価格 13,800円(税別)

マウス
CZ-8NM2A
標準価格 6,800円(税別)

ジョイカード
CZ-8NJ1
標準価格 1,700円(税別)

その他

拡張スロット

拡張 I/Oボックス(4スロット)
(CZ-600C/601C/611C/602C/612C/603C/613C/623C用)
CZ-6EB1-BK
CZ-6EB1
標準価格 88,000円(税別)

スピーカー

アンプ内蔵
スピーカーシステム(2本1組)
AN-S100
標準価格 36,600円(税別)

システムラック

システムラック
(CZ-600C/601C/611C/602C/612C/603C/613C/623C用)
CZ-6SD1
標準価格 44,800円(税別)

※4 ご使用に際しては、あらかじめ別売の1MB増設RAMボードCZ-6BE1 標準価格35,000円(税別・CZ-600C用)、CZ-6BE1B 標準価格28,000円(税別・CZ-601C、CZ-611C、652C、653C、662C、663C用)を増設してください。
※5 モデムユニットCZ-8TM2に同梱のソフトはX1/X1ターボシリーズ用です。

品名	品番	価格
●ミニフロッピーディスクユニット(2D)	★CZ-502F	99,800円
●ミニフロッピーディスクユニット(2D・1ドライブ)	CZ-503F	49,800円
●増設用ミニフロッピーディスクドライブ(2D)※4	CZ-53F-BK	19,800円

拡張ボード・その他

品名	品番	価格
●モデムユニット(300/1200ボー)	CZ-8TM2	49,800円
●320KB外部メモリ	CZ-8BE2	29,800円
●RS-232C・マウスボード※5	CZ-8BM2	19,800円
●フロッピーディスクインターフェイス※6	CZ-8BF1	14,800円
●JIS第1水準漢字ROM※7	CZ-8BK2	19,800円
●RS-232C用ケーブル(平行接続型)	CZ-8LM1	7,200円
●RS-232C用ケーブル(クロス接続型)	CZ-8LM2	7,200円
●拡張 I/Oボックス	CZ-8EB3	33,800円
●RFコンバータ※8	AN-58C	2,980円
●インテリジェントコントローラ	CZ-8NJ2	23,800円
●マウス・トラックボール	CZ-8NM3	9,800円
●マウス	CZ-8NM2A	6,800円
●トラックボール	CZ-8NT1	13,800円
●ジョイカード	CZ-8NJ1	1,700円
●チルトスタンド	CZ-6ST1-E・-B	5,800円
●高性能CRTフィルター ※9	BF-68PRO	19,800円
●スキャナ用パラレルボード ※10	CZ-8BN1	27,800円

●品番中の－表示は、B〈ブラック〉・E〈オフィスグレー〉を示します。※1 X1ターボZシリーズ用 ※2 CZ-862Cには接続できません ※3 X1ターボシリーズ用 ※4 CZ-830C用 ※5 X1シリーズ用 ※6 CZ-850CでCZ-520Fを使用する場合に必要 ※7 CZ-800C、801C、802C、803C、811C、820C用 ※8 CZ-820C、822C、830C用 ※9 14/15型用 ※10 CZ-8NS1用 ●接続等の説明につきましては、周辺機器総合カタログをご参照ください。

★印の商品は在庫僅少です。

本広告に掲載しております商品および役務の価格には消費税は含まれておりませんので、ご購入の際、消費税額をお支払い下さい。

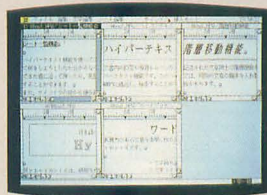
電子機器事業本部液晶映像システム事業部第2商品企画部 〒162 東京都新宿区市谷八幡町8番地☎(03)260-1161(大代表)

SHARP

"アート"と呼べる高水準のソフトウェアが

並のワープロじゃものたりない。アイデアあふれる人の知的ツール、「ハイパーワード」新登場。

X68000の優れたグラフィック環境をユーザーインターフェースに活用して、効率的に文書を作成するためのインテリジェントツール、それがハイパーワードです。"WYSIWYG"な画面で表現力あふれる文書を作成、印刷できます。アイデアをうまく活用できるアイデアプロセッサ機能、論文やメモ、個人ノートなどを有機的に結合、検索できるハイパーテキスト機能をサポート。データの整理に、プレゼンテーションツールに、単なるワープロを超えた幅広い用途に利用できます。

《ワードプロセッサ機能》●4種類の文字サイズ●9種類の文字修飾、4種類の回転、8種類の下線、8種類の罫線●4種類の割り付け●英文編集機能装備

ハイパーテキストワープロ PRO-68K
Hyperword
CZ-251BS 標準価格39,800円(税別)

情報人の24時間をマネージメント、データと上手につきあえるサイバーノート。

プライベートなデータやビジネスデータを簡単な操作で管理・運営できるパーソナルデータベースです。リフィル、タックシール、ハガキなどへの印字もOK。シャープ電子手帳とのデータ交換(別売の通信ケーブル CE-200Lが必要)も実現。電子手帳をX68000の情報端末として利用できます。

CYBERNOTE PRO-68K
CZ-243BS 標準価格19,800円(税別)

必要なときいつでも使える、メモリ常駐型のステーショナリーツール。

他のソフトを起動する前に、このStationeryPRO-68Kを一度起動するだけ。これで、他のソフトを実行中にも、「メモ」や「スケジュール」、「住所録」など、このソフトが持つ多彩な機能がワンタッチで使えます。またシャープ電子手帳とのデータの送受信も実現(別売の通信ケーブル CE-200Lが必要)。

Stationery PRO-68K
CZ-240BS 標準価格14,800円(税別)

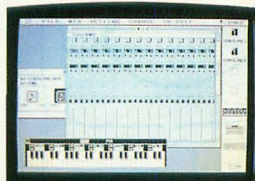
シャープ株式会社 ●お問い合わせは…シャープ㈱電子機器事業本部液晶映像システム事業部第2商品企画部 〒162 東京都新宿区市谷八幡町8番地 ☎(03)260-1161(大代表)へ。

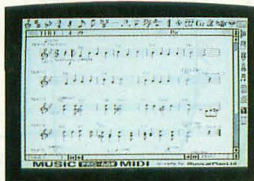
資料請求券 CZソフト oh!X 6体

X68000をサポート。

シャープオリジナルソフトウェア
X68000

サウンドツール

Musicstudio PRO-68K ver.1.1
■CZ-252MS 標準価格28,800円(税別)
24トラック対応MIDIマルチレコーディングソフトMusicstudio PRO-68Kがバージョンアップしました。従来の機能に加え、小節間のコピー及びデリートや、MIDIインプットモニターなど、数々の機能を追加・改良。さらに使いやすくなりました。
※MIDIボード(CZ-6BM1)が必要です。

MUSIC PRO-68K〔MIDI〕
■CZ-247MS 標準価格28,800円(税別)
MIDI対応自動伴奏機能をサポート、簡単な楽譜入力で演奏が楽しめます。
※MIDIボード(CZ-6BM1)が必要です。

ソングライブラリ〈101曲集〉
■CZ-248MS 標準価格8,800円(税別)
鑑賞用と音楽データ加工作成用からなるライブラリです。

Sampling PRO-68K
■CZ-215MS 標準価格17,800円(税別)
AD PCM機能を活かす高機能サンプリングエディタ。多彩なEDITORを装備、サンプリング音のデータはBASICでも活用できます。

SOUND PRO-68K
■CZ-214MS 標準価格15,800円(税別)
スタジオのコンソールパネルを操作する感覚でFM音源による音創りが楽しめるサウンドエディタ。

MUSIC PRO-68K
■CZ-213MS 標準価格18,800円(税別)
最大8パートのスコア(総譜)が書け、内蔵のFM音源で演奏できる楽譜ワープロ&演奏用ツール。

アートツール

NEW PrintShop PRO-68K
■CZ-221HS 標準価格19,800円(税別)
オリジナリティあふれるはがき等、簡単に作成、印刷できるホームプロダクティビリティツール。ほとんどの処理をアイコンで表示しマウスで選ぶフレンドリーオペレーション。

グラフィックライブラリ VOL.1
■CZ-235GS 標準価格8,800円(税別)
暑中見舞用を中心としたNEW Print Shop PRO-68K用グラフィックデータ集。

グラフィックライブラリ VOL.2
■CZ-236GS 標準価格8,800円(税別)
年賀状を中心としたNEW Print Shop PRO-68K用グラフィックデータ集。

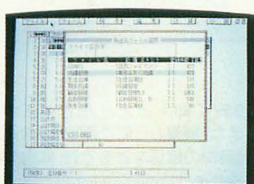
ビジネスツール

TOP給与計算エキスパート
■CZ-228BS 標準価格200,000円(税別)
給与計算から明細発行までを、リアルイメージ入力により自動的に、素早く処理することができます。

TOP財務会計
■CZ-227BS 標準価格200,000円(税別)
会計エキスパートシステムとデータベースを搭載し、機能と操作性を両立させた財務会計ソフト。

CARD PRO-68K
■CZ-226BS 標準価格29,800円(税別)
自由なレイアウト画面で入力できるワープロ機能を装備したカード型リレーショナルデータベース。

CARD PRO-68K用システム手帳リフィル集
■CZ-241BS 標準価格9,800円(税別)

CARD PRO-68K用活用フォーム集
■CZ-242BS 標準価格9,800円(税別)

DATA PRO-68K
■CZ-220BS 標準価格58,000円(税別)
コマンド入力の手間を軽減するヒストリー機能、罫線ドライバー付レポートライター機能、10進31桁の高精度演算。さらにイメージ表示機能を装備したコマンド型リレーショナルデータベースです。

BUSINESS PRO-68K
■CZ-212BS 標準価格68,000円(税別)
スプレッドシート(表計算)、データベース、グラフ作成機能を緊密に一体化させた統合ビジネスツールです。マウス対応のやさしいオペレーション、高度なエディタ機能、豊富な関数群など、初心者からプロまで幅広く使えます。

開発ツール

OS-9/X68000
■CZ-219SS 標準価格29,800円(税別)
X68000のもつグラフィック環境はもちろん、AD PCM音声、FM音源とグラフィックの同時再生といったマルチメディア機能をサポート。OS-9のもつマルチタスク機能、リアルタイム機能を活かした使い易く機能的なOS環境を提供します。また、これまでのデータ資産も活かせます。 ※OS-9はマイクロウェア社の登録商標です。

C compiler PRO-68K
■CZ-211LS 標準価格39,800円(税別)

Human68k ver2.0
■CZ-244SS 標準価格9,800円(税別)

THE福袋V2.0
■CZ-224LS 標準価格9,980円(税別)

AI-68K(Staff LISP/OPS PRO-68K)
■CZ-234LS 標準価格188,000円(税別)

通信ツール

Communication PRO-68K
■CZ-223CS 標準価格19,800円(税別)
300〜19,200BPSまでの通信速度に対応し、各種データベースの漢字端末やパソコン通信に利用できる高機能通信ソフトです。逆スクロール機能や自動実行機能、また豊富な編集機能を装備。

シューティングゲーム
〈ツインビー〉
■CZ-217AS
標準価格7,800円(税別)
©KONAMI. 1988

シューティングゲーム
〈沙羅曼蛇〉
■CZ-218AS
標準価格8,800円(税別)
©KONAMI. 1989

ブロックゲーム
〈アルカノイド〉
■CZ-222AS
標準価格7,800円(税別)
©TAITO CORP. 1987

ドライブゲーム
〈フルスロットル〉
■CZ-231AS
標準価格8,800円(税別)
©TAITO CORP. 1988

スポーツゲーム
〈熱血高校ドッジボール部〉
■CZ-232AS
標準価格7,800円(税別)
©TECHNOS JAPAN CORP. 1988

アクションゲーム
〈パックマニア〉
■CZ-233AS
標準価格7,800円(税別)
©NAMCO

アクションゲーム
〈ニュージーランドストーリー〉
■CZ-230AS
標準価格8,800円(税別)
©TAITO CORP. 1989

スポーツゲーム
〈V'BALL〉
■CZ-246AS
標準価格7,900円(税別)
©TECHNOS JAPAN CORP. 1989

バイクレーシングゲーム
〈スーパーハングオン〉
■CZ-238AS
標準価格8,800円(税別)
©SEGA 1987

ジェットヘリ・シミュレーションゲーム
〈サンダーブレード〉
■CZ-239AS
標準価格9,500円(税別)
©SEGA 1987

アクションゲーム
〈ダウンタウン熱血物語〉
■CZ-254AS
標準価格8,800円(税別)
©TECHNOS JAPAN CORP. 1989

本広告に掲載しております商品および役務の価格には消費税は含まれておりませんので、ご購入の際、消費税額をお支払い下さい。

資料請求 No.106

誕生!
〈NEW VERSION〉
Ver. 3.0

ボクセル分割による高速化。
アニメーション機能の充実。
マッピング環境の充実。

C-TRACE98 Ver. 3.0
(PC-9801シリーズ対応)¥98,000

C-TRACE68 Ver. 3.0
(X68000シリーズ対応)¥98,000

バージョンアップ受付中。

68ユーザーへお知らせ
アニメーションツールボックス「うごくZO」ARGOより発売
価格¥7,800(税込み)C-TRACEクラブ会員価格¥5,800(税込み)
問い合わせTFL.03-5996-4459

98ユーザーへお知らせ
オプション創像・写像ペイント セットで¥39,800

C-TRACE TOWNS	¥68,000
C-TRACE 98+(PC-9801シリーズ)	¥198,000 近日発売
C-TRACE NEWS Ver.3.0(SONY)	¥530,000
★C-TRACE 98TP	¥610,000
★C-TRACE 68TP	¥610,000

表示価格に消費税は含みません。
★の製品は店頭販売いたしておりません。直接当社までお申し込みください。

CasT

株式会社キャスト
●お問い合わせ先●
〒158東京都世田谷区等々力2-1-13
TEL.03-705-1065 FAX.03-705-5224

X68000

本格的ファイルマネージングソフトウェア

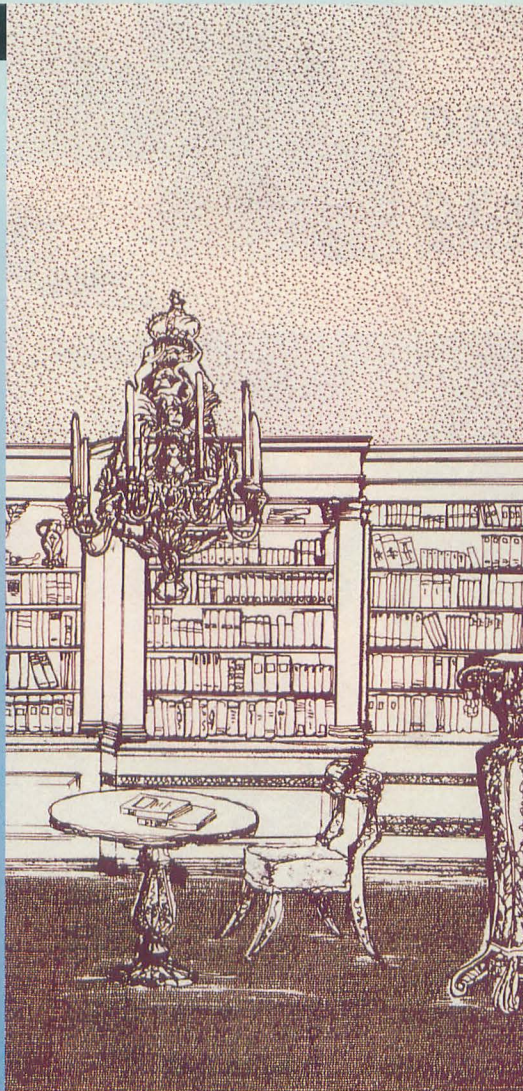
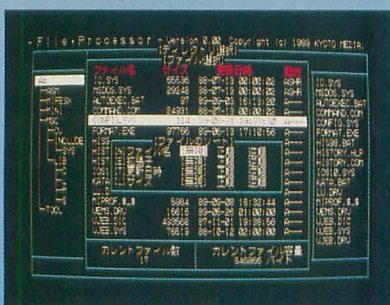
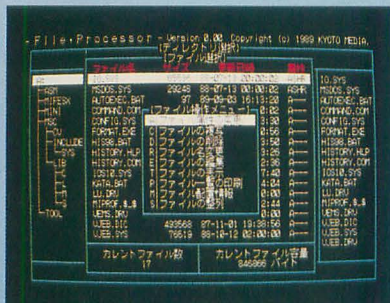
業界の新星、ロゴスシステムが
ユーザーの希望を1つの形にしました。
これは必要だとか便利じゃない、快感だ！

全国有名パソコンショップでお求め下さい。
電話1本での通信販売も受付いたしております。

THE FILE PROFESSORの実力

ディスクのバックアップ、ディスクのエディット、ディスクの初期化、ディスクの比較、ディスクの検査、ディスクの情報、FATのエディット、ファイルの検索、ディレクトリのコピー、ディレクトリの削除、ヴォリュームラベルの設定、ディレクトリの作成、ディレクトリ構造の再読み込み、ディレクトリ構造の印刷、ディレクトリ名の変更、ディレクトリ内容のソート、削除ファイルの復元、ファイル属性の変更、ファイルのコピー/移動、ファイルの削除、ファイルのエディット、ファイルの配置情報、ファイル一覧の印刷、ファイル名の変更、ファイルのソート、ファイル更新日時の変更、ファイルの表示、ファイルの奨行、カレンダー、ハードディスクの直接エディット、システム情報の表示、コマンドシェル、現在時刻の変更。

メニュー選択方式を実現!!
初心者でも簡単に使える

（画面写真は、98用を開発中のものです。）

The File Professor

ロゴスシステム

このソフトはロゴスシステムのデビュー作です。でも、だからといってなめてもらっちゃぁ困ります。私達は、いろいろなソフトを作りました。そのどれもが他社から発売されていました。出来る事ならば自分達で発売したい！その願いがやっとかないました。

好評発売中！

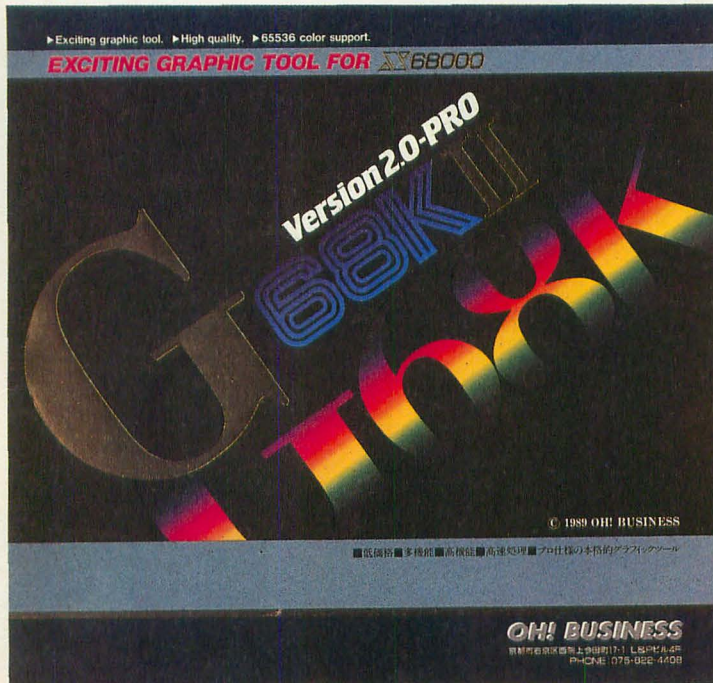
定価28,000円

ロゴスシステム
〒615 京都市右京区西院上今田町17-1 L&Pビル4F
TEL (075) 812-6383　FAX (075) 822-6915

G68K Version II-PRO 登場!

NEWS!

OH! BUSINESS
●京都市山科区音羽西林町2
サポート室:(075)502−2972
開　発　室:(075)822−4408

▶Exciting graphic tool. ▶High quality. ▶65536 color support.
EXCITING GRAPHIC TOOL FOR X68000
Version 2.0-PRO
G68K II Tool
© 1989 OH! BUSINESS
OH! BUSINESS

発/売/中/!/

G68K Version II-PRO

定価:¥22,000

ご　案　内

この度、弊社では発売中のG68Kをバージョンアップ致しましたので、下記のとうりご案内させていただきます。

旧版G68Kは、お求めやすい価格と簡単操作により、入門用ツールとして多くのX68000ユーザーの皆様方よりご好評をいただいております。

今回のバージョンアップでは旧版の簡単操作を継承しつつ、業界でもトップレベルの処理スピードと前作を遥かに上回る、高機能・多機能・高速処理を実現致しました。

旧版G68Kユーザーの皆様方から頂いた多くのご意見を元に、本格的プロ仕様ツールとして大幅バージョンアップ致しました。

サンプルデータもプロのイラストレーターの手に依るコンピュータイラストを収録。また、専用グラフィックデータ集のシリーズ化、発売を近日中に予定いたしております。

高速・高機能・低価格・1MB標準実装のメモリで完全に動作する本格派グラフィックツール。

■前作を大幅に上回る80種類のパレット
- 自由に編集可能
- 模様のついたパレットも作成可能
- HSV方式による色の合成
  色相(色の種類)・彩度(色の濃さ)・明度(色の明るさ)
- 簡単にお望みの色を作り出すための数々の機能を装備
- マスキング塗料・マスク除去塗料を装備
  微妙な修正に威力を発揮
- 2色の混合
- 画面上より自由に色を取り込むスポイト機能
- パレット保存可能
- 画面上より自由にタイルパターンを取り込むタイルパターン用カッターを装備

■32階調の濃淡をもつブラシ
- 自由に形状を変更できるブラシが24種類
- ユーザーが自由に変更・ディスクに保存可能

■大幅に機能アップされたエアブラシ
- ブラシノズル口径、インク噴出速度・濃度を自由に設定

■32階調の濃淡を持つトーンパターン
- 全てのペイントに有効
- 自由に変更・ディスクに保存可能

■強力な編集機能
- 2倍、4倍、8倍に画面を拡大する拡大エディット機能(ルーペ機能)
- 色を調整するカラーコレクタ
- 任意角度の高速画像回転
- 拡大・縮小
- 左右・上下反転
- 切り取りセーブ&ロード
- 自由領域のコピー・移動
- 標準実装のメモリで全画面が編集可能

■製図用具
- マスキング機能
- ペン描画時の直線
- 指定領域のカラー変更
- 円・楕円・ボックス・直線・自由領域
- これらの内部のペイント
- 単色領域ペイント

■文字入力をサポート
- X68000標準24×24ドットキャラクタの表示

■外部機器のサポート
- 豊富な対応周辺機器など　●各種プリンター・イメージスキャナ・カライメージユニット他

■起動直前の画面を保存しながら起動することも可能

■UNDO機能(取り消し処理)
- ペイント等に失敗してもワンステップ前に戻ることが可能

■市販グラフィックツールとのファイルコンバーターが付属
- Z's STAFF-PRO 68Kとのファイル変換が可能

■ノンプロテクト
- ハードディスクへの転送も可能(自由インストール)

■FileはBASICのGL3形式
- BASICより簡単に読み出し可能

▶お問い合わせ・お申し込みは上記電話番号までお願い致します。(上記サポート室迄)

MIDI対応
ジェミニウイング
Gemini Wing
TM

艶やかなファッション界を襲う奇怪な連続殺人事件。
南米の血に隠された秘密とは?
そして魅由を待ち受ける血族の宿命は?

あたし、魅由。
新宿にあるデザイン・スタジオの、新人A・D(アパレル・デザイナー)。……なんだけど、あたしの持ってる妙な「力」みたいなモノ—— 人の心が判っちゃったり、変にカンが良かったり——のせいで、周りからは「名探偵魅由」なんて呼ばれて、よく相談事を持ち込まれたりしている。で、そんなある日、友達のモデルが、突然、殺されてしまった。そして、あたしの親友だった唯も……!
これって……ひょっとして連続殺人事件ってヤツ?!

美少女名探偵 魅由の繰り広げる
ミステリアスアニメーションアドベンチャー第1弾!!

新発売!!

X68000対応 5"-2HD
標準価格8,800円

・ローランド社
MT-32完全対応

(MIDIインターフェイスボードC-Z-6BMI
又は、SACOM製SX-68Mが必要です。)
注)初期のMT-32では、正常に演奏できません。

X68000 SERIES

MIDI INTERFACE BOARD
SX-68M

体感 快感 実感

純正コンパチブル

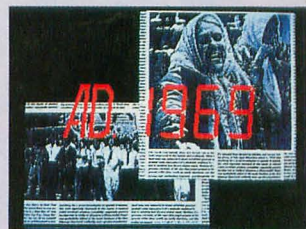
「SX-68M」は、純正品との互換性を保ちつつ(※)さらに、お求めやすい低価格におさえた、X68000シリーズ専用MIDIインターフェースボードです。
特徴としては、ボード本体に直接MIDI規格のDINコネクタを装着することによって、中間に変換ケーブルを使用する必要がなくなりました。また、クロック部に安定度の高いオシュレーターを採用することにより、さらに信頼度の高いものとなっております。もちろん、従来のMIDIボードをサポートするソフトウェアはそのままお使いになれます。SX-68Mで、あなたもすばらしいMIDIの世界を体験してください。

(※)本ボードは、TAPE SYNC端子を装備していないため、その機能をサポートするソフトは、ご使用いただけません。また、本ボードは、2枚同時装備ができませんので、ご注意ください。

SX-68M仕様

品名	MIDIインターフェースボード
規格	MIDI規格 1.0準拠
コントロールLSI	日本楽器(YAMAHA) YM3802
MIDI端子	MIDI OUT 2端子 MIDI IN 1端子
	MIDI OUT 1端子 MIDI THRU 1端子 MIDI IN 1端子
電源	+5V 170mA(本体より供給)
外形寸法	150mm(W)×167mm(D)×23mm(H)
重量	約160g

標準価格 ¥19,800

対応ソフト紹介

■38万キロの虚空

■メタルサイト

株式会社 システム サコム
〒130 東京都墨田区両国4-38-16 両国桜井ビル4F
ハードウェア部 TEL.03(635)5145
ソフトウェア部 TEL.03(635)7609

38万キロの虚空
CD.東芝EMIより
7月発売

※標準価格には消費税は含まれておりません。

■発売 ビクター音楽産業株式会社

通信販売 当社の商品をお近くのパソコンショップでお買い求めになれない場合、商品名、機種名、住所、氏名、電話番号を明記のうえ、下記住所まで定価プラス3%消費税分を現金書留にてお申し込み下さい。(送料無料) 〒151 東京都渋谷区千駄ヶ谷2-8-16 ビクター音楽産業㈱〈通信販売係〉

ワンダラーズ フロム イース［イースIII］

WANDERERS FROM YS

By Falcom

X68000の為の書き下ろし32曲（新曲6曲）。FM音源とADPCMの絶妙なバランスでくり出す美しいBGMにのせて、高速三重スクロール＋横スクロールで描く遠近感にあふれるグラフィック。またー つ、ゲームソフトの神話が生まれた。　（ジョイスティック対応）

In my time, I've wandered everywhere
Around this world, Hope would always be there

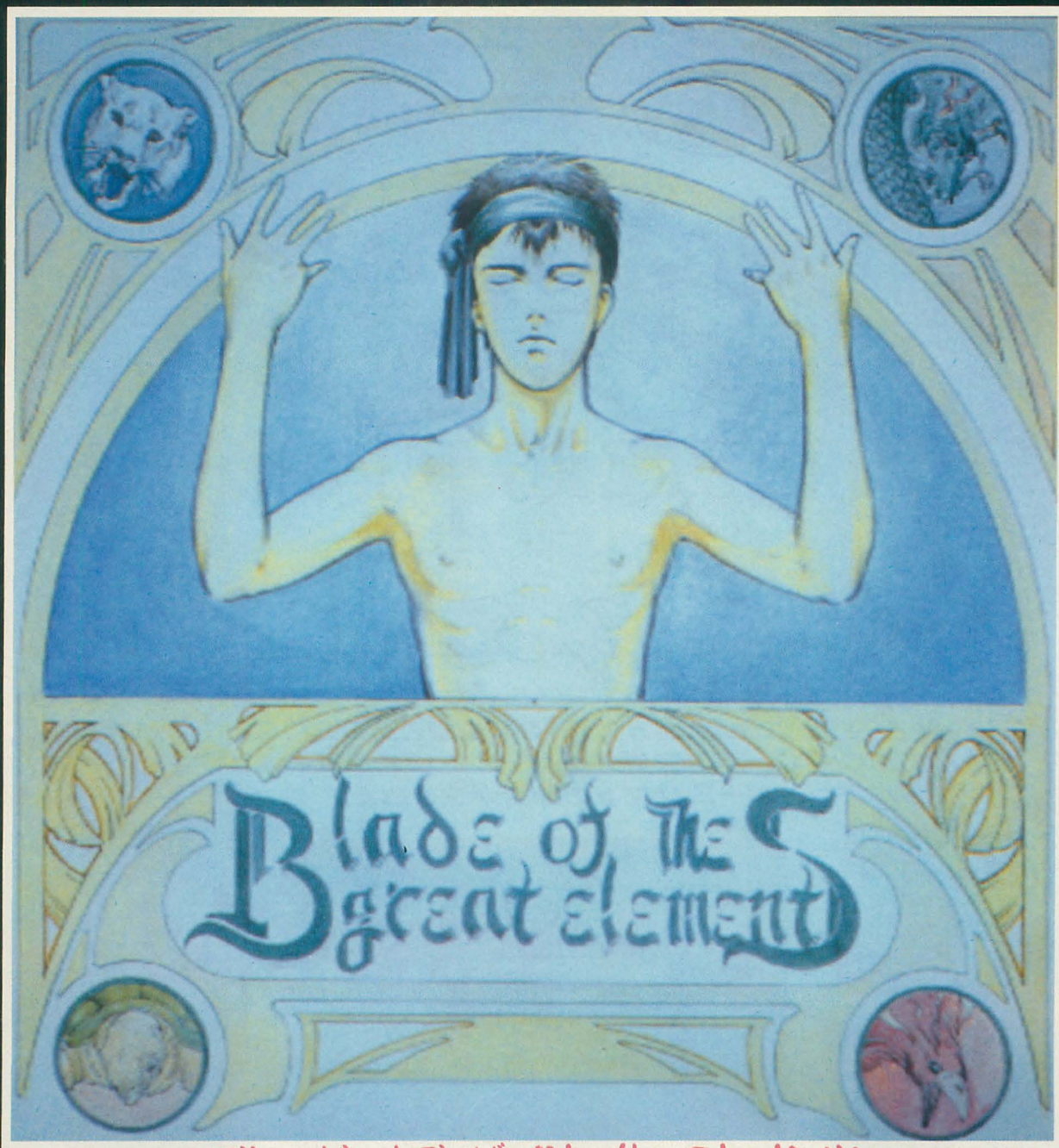
5′2HD（4枚組）価格8,700円

好評発売中!!

通信販売　（送料無料）

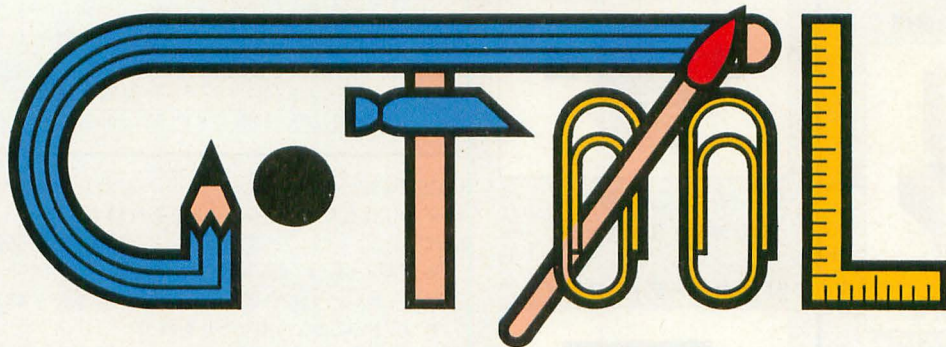
● 代金引換の場合
電話やFAXやハガキで品名・機種名・住所・氏名・年齢・電話番号を明記して申し込み下さい。商品お届け時に商品代金お支払い下さい。

● 現金書留の場合
品名・機種名・氏名・電話番号を明記して現金書留で申し込み下さい。

〒190 東京都立川市柴崎町2-1-4 ☎0425(27)6501

6月下旬
発売予定

グラフィック彩前線

オリジナルゲームを作成しようという方に使ってほしい、
本格派グラフィックツール、「Gツール」(X-68000対応)新登場。

GRAPHIC SPRITE CHARACTER EDITOR

〈主な機能〉

- 最高512×512ドットまでのスプライト、BG、グラフィックキャラクタの作成可能。
- 処理枚数は最高12枚。
- ウインドゥシステムにより、必要なアイコンは全て画面上に置くことができるため、スムーズな作業が可能。
- 使いたい機能は、マウスの左右のボタンに設定可能。
- スプライト、グラフィックキャラクタの作成には、ペン、タイル、ライン、ボックス、グラデーションフィル……の機能が使用可能。
- 文字入力可能。
- マスク機能
- 登録した図形を自在に表現できるイメージ入力機能。

¥28,000

ZAINSOFT
株式会社ザイン・ソフト
〒676 兵庫県高砂市米田町米田1162-1
TEL.(0794)31-7453

資料請求券
X 6月号

■店頭にて、ゲームソフト25％OFF!!（税別）、超低金利 ハッピークレジットをご利用ください!!
■特に人気のある商品によっては、しばらくお待ち願うことがありますのでご了承下さい。

厳選された製品を、より安く、より早く、皆様のお手元に!!

広告掲載商品以外の製品も取扱っております。

チャンス！ X68000・SUPER（チタン）＝6月発売!!予約受付中!!　送料￥2,000

SX-WINDOW搭載。

●ザ・ワークステーションと呼ぶにふさわしいスーパーな68000.!! 新登場.!!
SUPER-HD。

※プレゼント！①MD-2HD10枚　②アフターバーナー（￥9,200）　③ジョイカード（連射式）　④シリコンキーボード（￥2,800）

X68000 SUPER-HD

●CZ-623C-TN＋CZ-613D-TN
定価合計￥633,000…大特価!! TEL下さい。

※マウス・トラックボール付!! ディスプレイにはスピーカ2個、チルト台付!!

他のディスプレイ①CZ-602D、②612D、③CZ-603D、④CU-21HDの組合せもございますのでお問い合せ下さい。

♡電話をかけて、ドンドン値切っちゃえ.!!
♡安くてゴメンなさい。今だけヨ.!!

※超低金利クレジットご利用下さい。1回～60回払い、頭金ナシ！ボーナス1回払い、ボーナス2回払いOK！

オクト面白GOODS!!

アイテック　（送料￥1,000）
X68000専用ハードディスク
アイテック
●X68000専用ハードディスク
◎IT-X640（定価￥158,000）
●40MB●アクセスタイム28ms
特価￥99,800
◎IT-X680（定価￥198,000）
●80MB●アクセスタイム20ms
特価￥129,800
限定

オクト特選　シャープ周辺機器　（送料￥1,000）

●CZ-6BE1　1MB増設RAMボード……（￥35,000）▶特価￥26,500
●CZ-6BE1A　1MB増設RAMボード……（￥38,000）▶特価￥28,800
●CZ-6BE2　2MB増設RAMボード……（￥79,800）▶特価￥60,500
●CZ-6BE4　4MB増設RAMボード……（￥138,000）▶特価￥104,800
●CZ-6BF1　増設用RS-232Cボード……（￥49,800）▶特価￥38,500
●CZ-6BG1　GP-IBボード……（￥59,800）▶特価￥45,000
●CZ-6BM1　MIDIボード……（￥26,800）▶特価￥20,500
●CZ-6BN1　スキャナ用パラレルボード‥（￥29,800）▶特価￥22,800
●CZ-6BP1　数値演算プロセッサボード（￥79,800）▶特価￥60,500
●CZ-6BO1　ユニバーサルI/Oボード…（￥39,800）▶特価￥30,500
●CZ-6EB1/BK　拡張I/Oボックス……（￥88,000）▶特価￥66,800
●CZ-6VT1/BK　カラーイメージ・ユニット…（￥69,800）▶特価￥53,000
●CZ-6BL1　LANボード……（￥268,000）▶大特価
●CZ-8NM2A　マウス……（￥68,800）▶特価￥5,300
●CZ-8NT1　マウストラックボール‥（￥98,800）▶特価￥7,500
●CZ-8NS1　カラーイメージスキャナ……（￥188,000）▶大特価
●CZ-6BC1　FAXボード……（￥79,800）▶特価￥60,500
●CZ-8TM2　モデムユニット……（￥49,800）▶特価￥38,000
●CZ-64H　増設ハードディスク…（￥120,000）▶大特価
●CZ-6TU GY/BK　RGBシステムチューナー…（￥33,100）▶特価￥25,000
●BF-68PRO　高性能CRTフィルター……（￥19,800）▶特価￥15,500
●SX-68M（システムサコム）　MIDIボード……（￥19,800）▶特価￥15,000
●PIO-68BE1-A（I/O DATA）　1MB増設RAMボード……（￥25,000）▶特価￥18,500
●PIO-6BE2-2M（I/O DATA）　2MB増設RAMボード……（￥50,000）▶特価￥37,000
●PIO-6BE4-4M（I/O DATA）　3MB増設RAMボード……（￥88,000）▶特価￥65,000

モデム・コーナー　（送料￥1,000）

オムロン
●MD-1200AIII…特価￥14,800
●MD-24FS4……特価￥31,500
●MD-24FS5……特価￥34,800
●MD-24FP4……特価￥27,900

熱転写カラー漢字プリンター（ケーブル用紙付）　送料￥1,000

CZ-8PC4　￥99,800

●48ドットサーマルヘッド
●B5～B4まで
●ハガキ可能
●カラー対応

大特価　オクト推選 TEL下さい！

①CZ-8PC3（24ドット熱転写カラー漢字プリンター）
定価￥65,800……特価￥45,000
②CZ-8PK9（24ピン漢字プリンター80桁）
定価￥89,800……大特価.!!TEL下さい。
③CZ-8PK10（24ピン漢字プリンター136桁）
定価￥97,800……大特価.!!TEL下さい。
④CZ-8PG1（24ピンカラー漢字プリンター80桁）
定価￥130,000……大特価.!!TEL下さい。
⑤CZ-8PG2（24ピンカラー漢字プリンター136桁）
定価￥160,000……大特価.!!TEL下さい。
⑥IO-735X（カラーイメージジェット）
定価￥248,000……大特価.!!TEL下さい。

パソコンラック　推奨　送料無料

①五段キャスター付

5段キャスター付
キーボードが収納できるから、手元でマウス操作がラクラクできる
棚板5段のマルチに活用できるディスク。
ウーン、こいつはデキル！
1325（H）×640（W）×700（D）
特価￥16,000

②四段キャスター付
4段キャスター付
どんなパソコンにもフレキシブルに対応！
使い易いデスクです。
1245（H）×614（W）×600（D）
特価￥12,000

X68000ソフト大セール実施中※ゲームソフトオール25％off

〈グラフィック〉●Z's STAFF PRO68K Ver.2.0
（シャフト）定価￥58,000
オクト特価￥40,000

〈データベース〉●KAMIKAZE
（サムシンググッド）定価￥68,000
オクト特価￥46,000

〈グラフィック〉●C-TRACE68
（キャスト）定価￥68,000
オクト特価￥51,000

〈C言語〉●C & Professional Pack
（マイクロウェアジャパン）定価￥58,000
オクト特価￥44,000

〈グラフィック〉●サイクロン エキスプレス
定価￥78,000
オクト特価￥58,000

●限定.!!
◉サイクロン
限定特価￥25,000
※＋￥20,000で、サイクロン エキスプレスに交換できます.!!

型名	商品	定価	特価
CZ-211LS	Ccompiler PRO-68K	￥39,800	￥28,800
CZ-212BS	BUSINESS PRO-68K	￥68,000	￥48,000
CZ-213MS	MUSIC PRO68K	￥18,800	￥13,500
CZ-214MS	SOUND PRO-68K	￥15,800	￥11,500
CZ-215MS	Sampling PRO-68K	￥17,800	￥12,800
CZ-219SS	OS-9 'X68000	￥29,800	￥21,000
CZ-220BS	DATA PRO-68K	￥58,000	￥41,000
CZ-221HS	New Print Shop PRO-68K	￥19,800	￥14,300
CZ-223CS	Communication PRO-68K	￥19,800	￥14,300
CZ-224LS	THE 福袋 V2.0	￥9,980	￥7,500
CZ-226BS	CARD PRO-68K	￥29,800	￥21,300
CZ-241BS	システム手帳リフィル集	￥9,800	￥7,500
CZ-242BS	活用フォーム集	￥9,800	￥7,500
CZ-244SS	Homan 68K Ver.2.0	￥9,800	￥7,500
CZ-247MS	MUSIC PRO-68K〔MIDI〕	￥28,800	￥20,800
CZ-240BS	Stationery PRO-68K	￥14,800	￥11,500
CZ-243BS	CYBER NOTE PRO-68K	￥19,800	￥15,200
EW		￥38,000	￥29,800
G-68K		￥14,800	￥11,400
E-68K		￥19,800	￥15,300

店頭ゲームソフトオール25％off！ビジネスソフト 25％より特価中

●尚、送料として1ケ￥500、2ケ￥700、3ケ以上で￥1,000となります。（税別）

★通信販売お申込みのご案内★　〒144 東京都大田区蒲田4-6-7 TEL：03-730-6271

お申込みはお電話でお願いします。お客様の〈住所〉〈氏名〉〈電話番号〉及び〈商品名〉をお知らせ下さい。●入金確認後ただちに商品をご送付いたします。

現金一括払い
銀行振込：お近くの銀行より（電信扱い）にてお振込み下さい。
現金書留：封筒の中に住所・氏名・商品名をご記入の上当社までお送り下さい。

クレジット
専用お申込用紙をお送り致します。ので、必要事項をご記入、ご捺印の上ご返送下さい。手続きは簡単です。

オクト ラクラク クレジット表

1回	2%	3回	2.5%	6回	3.5%	10回	5%
12回	5%	15回	7.5%	18回	9%	20回	10%
24回	11%	30回	14.5%	36回	15.5%	48回	20%

振込先
富士銀行　三菱銀行
久ヶ原支店　蒲田支店
㊌No.1824　㊌No.0278691
株式会社 億人（オクト）

※掲載の価格は変動しますので、まずは、お電話にてご確認ください。
※連休のお知らせ＝5/22㈫、23㈬は連休です.!!
※上記料金には、消費税は含まれておりません。消費税が付加されますので、詳しくは電話でお問合せ下さい。
※銀行振込、または、現金書留でご注文の際には、あらかじめ電話でご確認の上、お申し込み下さい。

■平成2年夏のボーナス一括払いOK!!（7月末）手数料ナシ!!おトクです。ぜひ!!超低金利クレジットをご利用下さい。

超低金利クレジットOK

またまた秋葉原でおなじみの

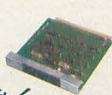
注目!!
夏のボーナス一括払い
手数料（金利）無料
（6月末・7月末どちらかご指定下さい）

5/15〜6/15

●お近くの方はお
●本体単品で特
●ビジネスソフト定

CYBER STICK

●CZ-8NJ2
（定価¥23,800）
超特価!!
▶価格はTEL下さい

X68000シリーズ専用
MIDIインターフェースボード
SX-68M（サコム）
（純生コンパチ）定価¥19,800
特価¥16,480
送料・消費税込み!

X-1ターボZⅢ 特別ご提供品!! 台数限定

●CZ-888C＋CZ-860D＋M-2HD（10枚）
定価¥269,600▶特価¥164,800
・ジョイカード
・ゲーム3種
・パソコンラックⒶ3段
プレゼント中
送料消費税込み!!
（ボーナス併用も有りますTEL下さい）

12回	14,400	24回	7,600	36回	5,300	48回	4,100	60回	3,400

ジョイスティック 送料¥500
●X-1PRO
定価¥9,500▶特価¥7,800
●ASCII STICK
定価¥6,800▶特価¥5,500

P&A超低金利クレジットをご利用ください!!

NEW X68000EXPERTⅡ/Ⅱ-HD & PROⅡ/PROⅡ-HD & SUPER-HD （送料・消費税込）

EXPERTⅡ

セットでお買い上げの方に、
●ディスケット10枚
●ゲーム3種
●ジョイカード2ケ
プレゼント中!!

EXPERTⅡ

Ⓐセット：CZ-603C＋CZ-604D……定価¥432,800▶特価（価格はお電話下さい。）

12回	28,400	24回	15,000	36回	10,300	48回	8,100	60回	6,700

Ⓑセット：CZ-603C＋CZ-605D……定価¥453,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓒセット：CZ-603C＋CZ-613D……定価¥473,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓓセット：CZ-603C＋CU-21HD……定価¥486,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

EXPERTⅡ-HD

セットでお買い上げの方に、
●ディスケット10枚
●ゲーム3種
●ジョイカード2ケ
プレゼント中!!

EXPERTⅡ-HD

Ⓐセット：CZ-613C＋CZ-604D……定価¥542,800▶特価（価格はお電話下さい。）

12回	35,800	24回	18,900	36回	13,000	48回	10,100	60回	8,500

Ⓑセット：CZ-613C＋CZ-605D……定価¥563,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓒセット：CZ-613C＋CZ-613D……定価¥583,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓓセット：CZ-613C＋CU-21HD……定価¥596,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

PROⅡ

セットでお買い上げの方に、
●ディスケット10枚
●ゲーム3種
●ジョイカード2ケ
プレゼント中!!

PROⅡ

Ⓐセット：CZ-653C＋CZ-604D……定価¥379,800▶特価（価格はお電話下さい。）

12回	25,200	24回	13,300	36回	9,200	48回	7,200	60回	6,000

Ⓑセット：CZ-653C＋CZ-605D……定価¥400,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓒセット：CZ-653C＋CZ-613D……定価¥420,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓓセット：CZ-653C＋CU-21HD……定価¥433,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	-?	48回	?	60回	?

PROⅡ-HD

セットでお買い上げの方に、
●ディスケット10枚
●ゲーム3種
●ジョイカード2ケ
プレゼント中!!

PROⅡ-HD

Ⓐセット：CZ-663C＋CZ-604D……定価¥489,800▶特価（価格はお電話下さい。）

12回	32,400	24回	17,000	36回	11,800	48回	9,200	60回	7,700

Ⓑセット：CZ-663C＋CZ-605D……定価¥510,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓒセット：CZ-663C＋CZ-613D……定価¥530,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓓセット：CZ-663C＋CU-21HD……定価¥543,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

SUPER-HD

セットでお買い上げの方に、
●ディスケット10枚
●ゲーム3種
●ジョイカード2ケ
プレゼント中!!

SUPER-HD

Ⓐセット：CZ-623TN＋CZ-604D……定価¥592,800▶特価（価格はお電話下さい。）

12回	39,800	24回	21,000	36回	14,500	48回	11,300	60回	9,500

Ⓑセット：CZ-623TN＋CZ-605D……定価¥613,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓒセット：CZ-623TN＋CZ-613D……定価¥633,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

Ⓓセット：CZ-623TN＋CU-21HD……定価¥646,000▶特価（価格はお電話下さい。）

12回	?	24回	?	36回	?	48回	?	60回	?

X68000シリーズ 〜P&Aスペシャルセット＝限定誌上販売!!

台数限定　送料、消費税込み

セットでお買上げの方に、
●ディスケット10枚 ●ゲーム3種 ●ジョイカード2個 プレゼント中

EXPERT

●CZ-602C＋CZ-612D……定価¥475,800▶特価¥306,000
●CZ-602C＋CZ-604D……定価¥450,800▶特価¥300,000
●CZ-602C＋CZ-605D……定価¥471,000▶特価¥320,000
●CZ-602C＋CZ-613D……定価¥491,000▶特価¥336,000
●CZ-602C＋CU-21HD……定価¥504,000▶特価¥338,000

EXPERT-HD

●CZ-612C＋CZ-612D……定価¥585,800▶特価¥375,000
●CZ-612C＋CZ-604D……定価¥560,800▶特価¥369,000
●CZ-612C＋CZ-605D……定価¥581,000▶特価¥386,000
●CZ-612C＋CZ-613D……定価¥601,000▶特価¥403,000
●CZ-612C＋CU-21HD……定価¥614,000▶特価¥407,000

PRO-HD

●CZ-662C＋CZ-612D……定価¥527,800▶特価¥339,000
●CZ-662C＋CZ-604D……定価¥502,800▶特価¥333,000
●CZ-662C＋CZ-605D……定価¥523,000▶特価¥352,000
●CZ-662C＋CZ-613D……定価¥543,000▶特価¥368,000
●CZ-662C＋CU-21HD……定価¥556,000▶特価¥372,000

●本広告の掲載の商品の価格については、消費税は含まれておりません。●営業時間＝平日AM10:00〜PM7:00、日祭AM10:00〜PM6:00

プロ、アマ、ヘタ、ウマ、

みんなおいで。

コンピューター グラフィックアート大募集

iam-NETは、CGアーティストや、CGアーティストを目指すキミたちの為につくられたネットワークだ。今回、iam-NETでは、そんなキミたちの夢のあるCGアートを募集。応募してくれた方には、審査の上、下記の賞品のほか、iam-NETに会員として無料登録の特典もある。これまでのイメージにとらわれないエネルギーあふれる作品を待っている。さあ、はりきって応募しよう!!

賞/グランプリ(1名)、奨励賞(6名) 1次締切分 3名 2次締切分 3名

(グランプリ)シャープカラー イメージスキャナ、CZ-8NS1、もしくは、ローランド音源ボードCM-64プラスMIDIボードCZ-6BM1

(奨励賞)AIWAモデムPV-A24V42

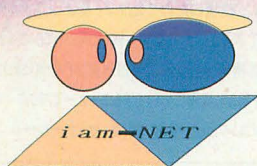
(会員に登録された方全員にiamオリジナルTシャツプレゼント)

※応募された方には個々に仕事を御願いする場合、別途に仕事の内容、契約事項等を御送りさせて頂きます。

テーマ

①あなたの自由なオリジナルテーマ
②古典名画をCGでパロディ(ex、モナリザ)
③幾可学文様、抽象的グラフィック、マンダラ的文様等
(3Dグラフィックも勧迎。)

締切

一次締切 平成2年6月20日㈬消印有効
二次締切 平成2年7月20日㈮消印有効

i am-NET
COMPUTER ARTIST NET WORK

応募方法

- 1画面を1FDに納めて下さい。(バックアップFDと共に2FD)
- ファイル形式はGL3形式、Zim File形式、RGB File形式で御願いします。
- 画面数は問いません。
- 応募作品の入っているFDは返却できません。
- 申し込み用紙に必要項を必ず記入して下さい。
- 未発表の作品に限ります。
- その他著作権を侵害する作品は受付けられません。

応募先

大阪府堺郵便局私書箱195
iam-NET係

発表

商品の発送をもって発表にかえさせて頂きます。

住 所
氏 名 生年月日 年 月 日 性別 男・女
職 業 会社員・自由業(職種)・学生(大・高・中・小・専門学校)
TEL. FAX.
愛読雑誌 趣味 現在使用してる機種

応募ファイル名	タイトル名	作品のイメージ及びメッセージ

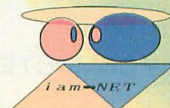
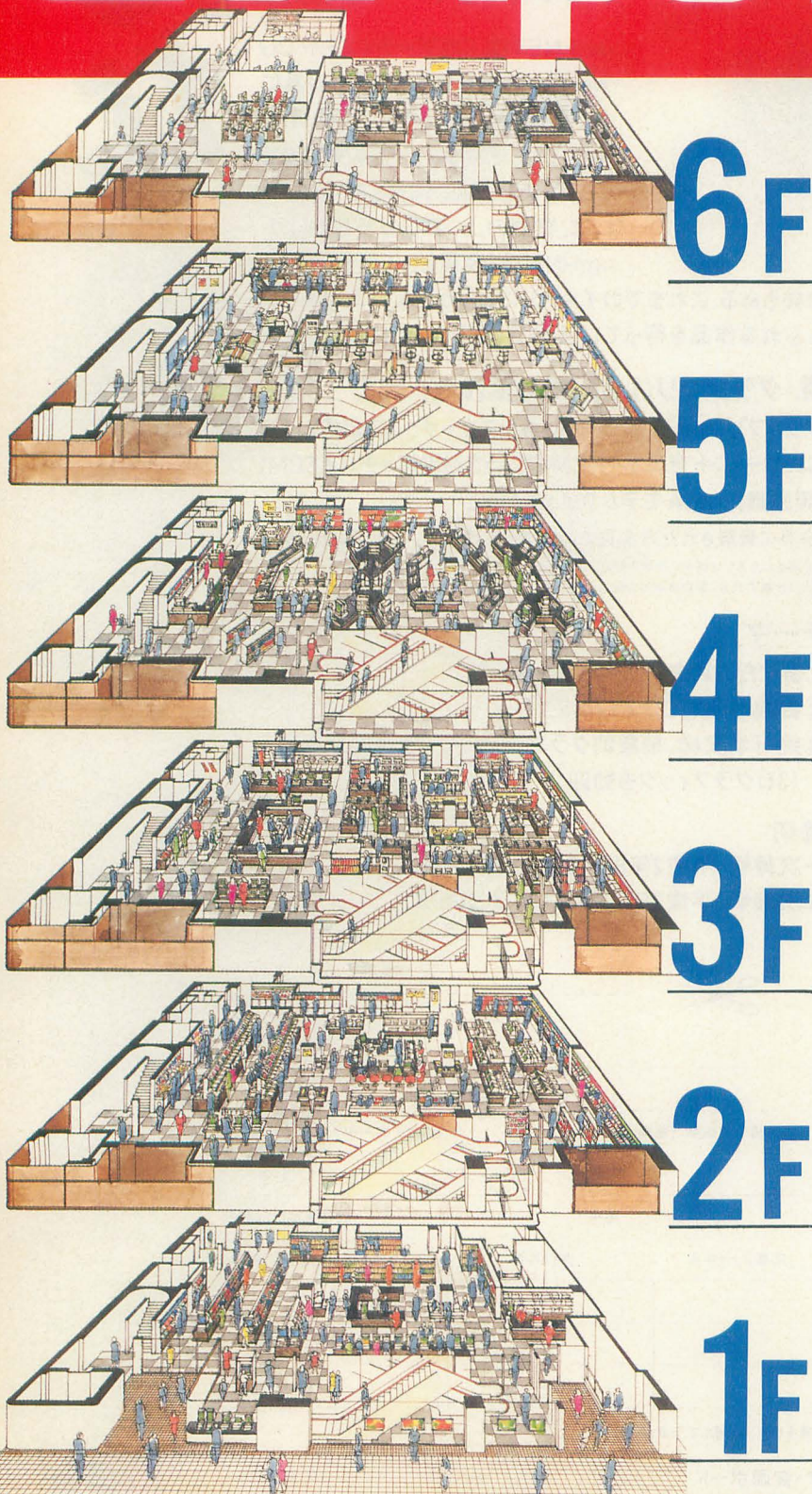
●グランプリ賞を1つ○で囲んで下さい。

スキャナ・音源ボード

iam individual art media お問い合せ TEL.0722-64-3770

THE COMPUTER館

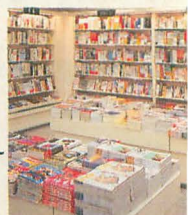
日本一のコンピュータ館です

6F
イベント フロア
催事場

5F
CAD&グラフィックス フロア
CAD・CAM・グラフィックスのソフトから周辺機器及びシステムまで/ワークステーション/LAOX NET/パソコン通信用機器及び体験コーナー

4F
ビジネスシステム フロア
業種別パソコンコーナー(小売業、不動産業、製造業、教育、広告、飲食店など業種にあわせたソフト及びシステムを動かし、実際に体験できるコーナーです)/業務別パソコンコーナー(社長室、営業部、総務部、経理部、OA推進部など業務にあわせたソフト及びシステムを動かし実際に体験できるコーナーです)/ラップトップパソコン/周辺機器/帳票/セミナールーム

3F
ビジネスソフト フロア
ビジネスソフト/帳票
IBM/J-3100/AX/Macintosh

2F
アクセサリー&ワープロ フロア
ワープロ/ワープロ周辺機器/電子手帳/OAシステムインテリア/コピー/ファクシミリ/アクセサリー/サプライ/セミナールーム

1F
ホビー&BOOK フロア
パソコン関連書籍・雑誌/ホビーパソコン・周辺機器/ゲーム・ホビーソフト/プレイゾーン

The COMPUTER館は、新製品のハードや周辺機器及びソフトをいち早く展示し、体験できます。

いつも楽しい催物や、役立つセミナーでいっぱい！ザ・コンピュータ館は目が離せません。

ダイナウェアDynaCADデビューフェア

パソコン用高速CAD「Dyna CAD」を体験できる3日間です。

●日時/5月18日㊎・19日㊏・20日㊐● 会場/ザ・コンピュータ館6Fイベントフロア

●体験Diskがもらえるスタ Dyna CAD入門セミナー/これからCADを使ってみたいという方にはいい機会です。Dyna CADを通じて、CADとは何かを学べ、終了後には、体験Diskまでもらえるうれしいセミナーです。

●自由にさわれる「Dyna CAD」体験コーナー/Dyna CADを実際に使って納得していただくために、会場はもちろんのこと、ご自宅でも自由に体験できるように、ワンポイントレクチャーを行ないます。

●レーザープリンタ出力・プロッタ出力ができるDyna CAD活用コーナー/実際にどのように活用できるかということで、建築プレゼンテーションでは、Dyna PERS3で、不動産プレゼンテーションではDESK UP2で作成したものを、出力します。

小型・A4ファイルサイズに高機能を搭載した、今話題のノートパソコンを知るチャンスです。

98ノート&ダイナブックセミナー

98ノート ●日時/5月25日㊎・26日㊏
ダイナブック● 日時/5月27日㊐・28日㊊
〉会場/ザ・コンピュータ館6Fイベントフロア

※セミナー受講のお申し込みや、詳しい内容については、当店セミナー係員にご遠慮なくお問い合わせ下さい。

THE COMPUTER「keyman USA」写真展

●日時/5月25日㊎～28日㊊ ●会場/ザ・コンピュータ館6Fイベントフロア

雑誌「THE COMPUTER」で紹介された、コンピュータ界の話題の人物を一同に集めた写真展です。どうぞお楽しみに！

MS-WINDOWSの世界

●日時/5月30日㊌～6月5日㊋ ●会場/ザ・コンピュータ館6Fイベントフロア

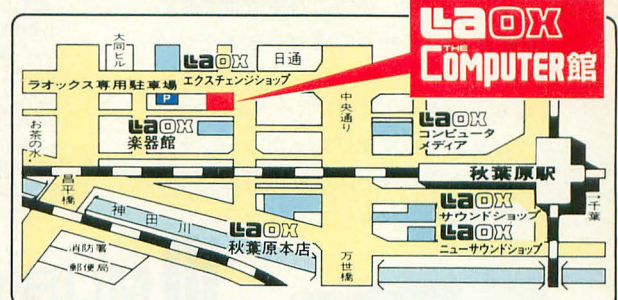
MS-DOSを基盤として構築された全く新しいOS環境であるMS-WINDOWS。我々にさまざまなメリットを与えてくれる、話題のMS-WINDOWSの世界を思う存分体験してみませんか。

ためになるパソコンソフトセミナーのご案内

●各セミナーとも専門の係員が、実際に皆さんと共にパソコンを起動させながら説明していきます。ぜひ、この機会をご利用ください。尚、各セミナーとも定員になり次第しめ切らさせていただきますのでご了承ください。お申し込み、お問い合わせは、お電話にて当店係員まで。多数のお申し込みが予想されますので、お早めにご予約ください。

月 日	時 間	セミナー名	ソフトハウス	ジャンル	フロア
5月18日(金)	17:00～19:00	PCA会計2	PCA	会 計	4F
5月19日(土)	14:00～16:00	Let'sアイリス	パーソナルメディア	データベース	4F
5月22日(火)	17:00～19:00	Aldus Page Maker	サムシンググッド	DTP	4F
5月25日(金)	17:00～19:00	弥生2	日本マイコン	会計(青色)	4F
5月26日(土)	14:00～16:00	Ninja3プロ	サムシンググッド	データベース	2F
5月27日(日)	14:00～16:30	ハードディスク入門	アイテック		4F
6月1日(金)	17:30～19:00	初めてさわるロータス1・2・3	ロータス	統合型表計算	4F
6月4日(月)	17:00～19:00	TOP財務会計(PCノート版)	オービック	会 計	4F
6月4日(月)	17:00～19:00	アシストカルク(PC9801用)	アシスト	表計算	2F
6月6日(水)	17:00～19:00	一太郎&花子(中級)	ジャストシステム	ワープロ&図形処理	2F
6月6日(水)	17:00～19:00	THE CARD3	アスキー	カード型データベース	4F
6月8日(金)	17:30～19:00	ロータス1・2・3(初級&中級)	ロータス	統合型表計算	4F
6月9日(土)	14:00～16:00	初めてさわるLet'sアイリス	パーソナルメディア	データベース	2F
6月10日(日)	14:00～16:00	マルチプランVer4.0	マイクロソフト	表計算	4F
6月13日(水)	17:00～19:00	桐	管理工学	データベース	4F
6月21日(木)	17:00～19:00	DYNA CAD & DYNA PERS3	ダイナウェアー	CAD	4F
6月27日(水)	17:00～19:00	MS-WINDOWS体験セミナー	日本ソフトバンク		4F
6月28日(木)	17:00～19:00	Aldus Page Maker	日本ソフトバンク	DTP	4F
6月29日(金)	17:00～19:00	マイクロソフトエクセル	日本ソフトバンク	表計算	4F

※イベントやセミナーの詳細につきましては、当店係員におたずねください。

その他、楽しく役立つイベントやソフトのデモ実演など最先端の情報がいっぱいです。

ザ・コンピュータ館

TEL/03-5256-3111

〒101 東京都千代田区外神田1-7-6(秋葉原駅徒歩4分) Ⓟ駐車場完備

シャープ見・体・験フェアin東京

4/14・15新宿NSビルB1F大イベントホール

もうすっかりお馴染みになったシャープ見・体・験フェア。東京地区の開催は4月14,15日の2日間で，前回のエルタワービルよりもさらに大きくメジャーな新宿NSビルの大イベントホールで行われた。なにしろ新製品のお披露目もかねた期待のイベントだ。もちろん入場は無料。例によってシャープからのダイレクトメールを受けたX68000ユーザーたちがどどっと押し寄せ，会場内はたちまち熱気の渦に巻き込まれた。

会場内では，新製品コーナーはもちろん，サードパーティのブース，名物ゲーム大会のコーナー，液晶ビジョンの部屋，講演会場，ミュージックステージ，そして人気のX68000グッズの即売会など楽しいコーナーがいっぱい。出展に参加した各ソフトメーカーのブースには発売されたばかりのEXPERTIIシリーズが並べられ金色のバッジを輝かせて，それぞれのオリジナルソフトを走らせる。また，シャープのコーナーでは，まだ店頭にも置かれていないX68000 SUPER-HDが初公開。チタンブラックってどんな色？　と詰め寄るユーザーの目をひいていた。当然，今回の目玉であるSX-WINDOWも見て触って確認できるというわけだ。

訪れる人の多くは実際にX68000ユーザーであったり，その友達であったり……というように，まさにユーザー同士の集いと呼べる2日間であったといえるだろう。

胸わくわく，期待でドキドキの僕たちを迎えてくれたのはこの看板でした。AXパソコンも展示されているようですね。はやる気持ちを抑えつつ熱気ムンムンの会場へ。

受付で紙袋を渡してくれるコンパニオンのおねえさん方。制服姿がステキです。「会場のパンフが欲しいんですけど」「申し訳ございません。今回は用意してありませんので後ろのパネルをご覧になってください」はい，とっても丁寧でした。

新機種発表！

新しいマン・マシン・インタフェイスSX-WINDOWが展示されていました。もちろん，見て触っての体験ができます。SX-WINDOWについては5月号と今月号を読んでください。熱心なユーザーに，コンパニオンのおねえさんが一生懸命答えていました。サンプルデータに森林林檎さんのグラフィックが使われているのはなぜでしょう。

X68000の最新機種の3台が展示されていました。やっぱり気になるのは，新色チタンブラックに塗られたX68000SUPER-HDみたいですね。パンフレットなどで見るとブラックとの区別がつきませんが，並べてみるとはっきり違いました。実物はとっても渋い色でかっこよかったと思います。見・体・験フェア中たった1台出品されたSUPERなのでした。

MUSIC

ローランドのブースでは，電子ピアノを使ったデモミュージックが流れていました。MIDIユーザーだけでなく，かなり関心が高かったようです。ローランドさんががんばるとX68000のMIDIの世界がおおいに広がることでしょう。

MIDIのライブステージでは，よだれものの楽器がところ狭しと並んでいました。もちろん，それらをコントロールするのは我らのX68000です。本番ではおねえさん2人も手伝ってニュースステーションのテーマや，名曲X68000のテーマなどが演奏され，ものすごい人だかりでした。これはセッティング風景です。

GRAPHIC

マジックパレットというよりスイカムシと言ったほうが通りがいいのはなぜでしょう。ミュージカル・プランのブースです。やっぱりスイカムシが大きな顔？　をしていました。

C-TRACEクラブではCGアニメをやっていました。6つものデモを用意しただけのことはあります。おもわず立ち止まってしげしげと画面を眺める人の姿が絶えませんでした。ところで，あのX68000には何Mバイトのメモリが載っていたんでしょうねえ。

ゲームソフト

今回も大画面ド迫力ゲームをやっていました。シャープお得意の液晶プロジェクターによる100インチ・スーパーハングオンです。ついでに大型扇風機でもあれば，まさに体感で完璧だったのに，なあんてつい思ったりして……。

ダンジョンマスターは，ここでもやっぱり人気があります。秋葉原では発売日に買えない人が続出したとか。常に誰かがマウスをころがしていました。でもこんな短時間じゃ最後まではいけないでしょうねぇ。

X68000とは切っても切れないS.P.S.のブースです。移植のうまさは電波新聞社に優るとも劣りません。サンダーブレードなどがデモっていました。その横ではC-TRACEのステゴちゃんが群棲していました。そばにいた女の子がおもわず「欲しいっ」とつぶやいておりました。

いきなりチェイスH.Q.で登場の日本デクスタのブースです。デモをやってなかったのが残念だけど，注目したいソフトハウスですね。

写真には出ていませんけど，コンパニオンのおねえさんが完全にシキっていたゲーム大会です。なかなか皆さん腕自慢のようですが，はたして勝利のコンパニオンは誰の頭上に微笑むのでしょうか。

あいかわらず電波新聞社のブースはすごい人だかり。バブルボブルがデモっていたところを見ると新作はなかったようですが，その代わりイースのキーホルダーなどが売られていました。ちなみに写真でプレイしているのは，バブルボブルのプログラマ君だったりします。

スタジオパンサーのブースでは，新作ゲームJOSHUAの兵器マニュアルとポスターが取り放題になっていました。全体的にポスターの人気が高かったようです。だって，シャープの紙袋からポスターがはみだしている人が多かったんですもの。

AX，ワープロ

おんや，Excelが走ってる？　と思いきやAXパソコンでした。X68000ほどではありませんが，たくさん並んでいました。このぐらいX68000もビジネスソフトが充実するといいですね。

会場の一郭では書院を使ったワープロの講習会も催されていました。ちなみに電子手帳の講習会もありました。

ソフト販売

シャープオリジナルグッズを展示即売していました。売れ筋は，テレカとフロッピータイトルシールなど安めのものに集中していましたが，Cコンパイラなども売り切れていました。それにしてもクリスタル・ポルシェってのは何物でしょう（私は前の見・体・験フェアで買ってしまったのですが）。

ソフトの即売会です。少し割引きしてソフトを売っていました。ブースを見ておもわず欲しくなり，帰りがけに買っていた人も少なくなかったのではないでしょうか。

ホットひと息 お疲れさまでした！

はぁ～，歩き疲れたわい，という人にはオレンジジュースやコーヒーがサービスされていました。会場の熱気はすさまじく，なんだかんだ言って3杯もおかわりをしてしまいました。アンケートに答えると文具ももらえたし，行って得する見・体・験フェアでした。

SOFTWARE INFORMATION

ダウンタウン熱血物語　あの熱血硬派くにおくんが活躍するアクションゲーム。日頃のうっぷんがこのゲームでいっぺんに晴らせそう。

SOFTWARE INFORMATION

今月はアクション，パズル，RPG，アドベンチャーと，あらゆるジャンルのゲームが出揃いました。夏休みに向けてか，各ソフトハウスでも秘密兵器を開発中らしいし，今後も目が離せない!?

話題のソフトウェア

すっかりあたたかくなって，まさに春たけなわといった今日この頃ですね。皆さんお元気ですかぁ？

さて，今月は紹介したいゲームがわりと多いので，さっさと進めていくことにしましょう。

まずはこの**ダウンタウン熱血物語**から。4月号でも紹介したこのゲーム，いよいよシャープから発売となりました。いわゆるケンカアクションゲームなのですが，チェーンをピシピシ，木刀ボコボコ，回転ジャンプで体当たりなど，なかなかそれらしくて楽しめます。また，お店の中でのくにおくんがかわいくて，思わずうれしくなっちゃいます。マップはわりと入り組んでいますが，そんなこたぁ気にせずビシバシやっていくのがいちばんです。ぜひプレイしてみてください。

お次は，ブロダーバンドジャパンの**パズニック**。いわゆる最近流行りのパズルゲーム（タイトル見りゃわかるか）ですがアクション性にも富んでいて，楽しめます。同じマークの駒を隣接させて消していくというもので，もともとはTAITOのアーケードものです。

そして，やっと画面写真をお届けできるようになったコナミの**クォース**です。こちらもパズルゲームですが，シューティングの要素も含んでいます。ゲームボーイやファミコンでも発売される予定なので，ちまたはクォースだらけになるかもね。

T&Eソフトでは，**ルーンワース～黒衣の貴公子～**が順調に開発が進んでいるようです。このRPGはプレイヤーの行動によっ

ほぉら，やっぱり日米欧の対決だ！

1.	ダンジョンマスター	1
2.	ポピュラス	6
3.	ワンダラーズ・フロム・イース	2
4.	バブルボブル	3
5.	アルガーナ	13
6.	ジェノサイド	9
7.	ソーサリアン	5
8.	ファーストクィーン	13
9.	A-JAX	7
10.	サンダーブレード	4

今回は，上位3作のしれつな争いが繰り広げられ，結局ダンジョンマスターが逃げきりました。このダンジョンマスターに関するアンケートハガキのコメントを読んでいると，月ごとにみんなゲームが進んでいく様子がわかって面白いなぁ。クリアしたぞ，というハガキもぽつぽつ届いてます。

一方，ダンジョンマスターに負けないだけのパワーはありながら，僅差で及ばず。ポピュラス。スタッフの間でも，対戦ポピュラスは人気のマト。西川善司対祝一平の試合の観戦チケットには，いまからプレミアムがついてるとかいないとか（ウソウソ！）。

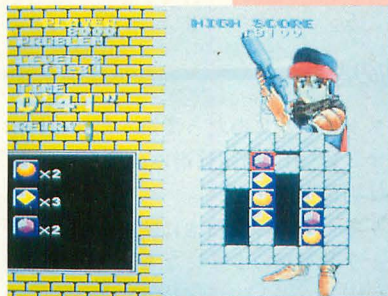
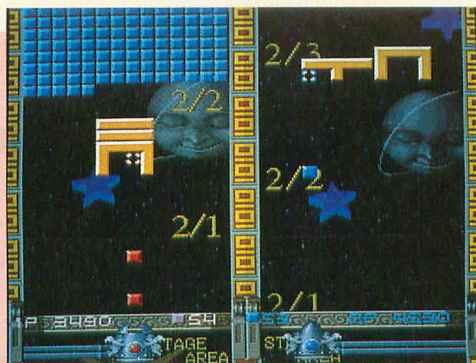
意外に（ゴメン）バブルボブルの人気が高いのにはびっくり。「奥が深い」「2人で遊べる」「デキがいい」などが主な支持理由かな。3強相手にどこまでねばるかに注目だ。

ねばるといえば，夏からずぅ～っと圏内でねばっているジェノサイドですが，CDの企画が進行中とか。あのハードなビートのアレンジバージョンというのも，聴いてみたいもんですね。ぜひ出してほしいなぁ。

今月は，アルガーナとファーストクィーンがそろって返り咲き。特にアルガーナは，「ランクインしてもらいたいから」というハガキまであるあたり，X1ユーザーのハガキにも気合いが入ってます。

圏外には，グラナダ，ファーサイドムーンという新作も控えて次回のランクインを狙ってますから，来月をお楽しみに。　（浦）

パズニック

クォース

ルーンワース～黒衣の貴公子～

ジェミニウイング

闇の血族

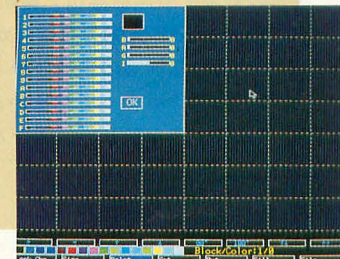
てストーリーが創られるそう。う～ん，わくわくしますねぇ。詳しいことは，情報が入り次第お伝えしていきますね。

先月号で紹介したシステムサコムの**ジェミニウイング**と**闇の血族**ですが，やっと皆さんに画面写真をお届けできるようになりました。どんなゲームかは先月号で簡単に書きましたが，両ゲームともMIDI対応とサコムとしても力を思いっきり入れているよう。完成が待たれます。

アルガーナ，LIFRAIMとたてつづけにゲームを出しているM.N.M.Softwareでは，なんと現在も数本の開発が進んでいるもよう。そのうちいくつかを紹介しましょう。まずビリヤードゲームの**POOL BAR**。このゲームは3Dで描かれ，視覚もプレイヤーの好みで変えられるといった凝りよう。仕上がりが楽しみです。そして，アドベンチャーゲームの**Vessel**。これはテキストアドベンチャーなだけにグラフィックはあまり入っていませんが，読んでいるうちにグイグイと世界に引き込まれそう。もうひとつは，ゲームじゃないけど，グラフィックツールの**ぴくせる君**です。実際にM.N.M.では，これを使ってゲーム作りをしているそうだから，出来のほどはうかがい知れますよね。そうそう，**PIPYAN**というパズルゲームも作っているんだそう。ほんとにたくさん作ってくれますよねえ，感心しちゃいます。あっと，人気の**アルガーナ**ですが，好評にお応えしてX68000版も開発中，そしてもちろんPart2も企画中とのこと。うれしいですね。これらは，順次発売していくそうですのでお楽しみに。

さて，そのほかのソフトをずらっと紹介してみるとしましょう。光栄からは**維新の嵐**のX68000版が発売されています。SPSではあの**ワールドコート**が順調に進んでいるよう。日本デクスタからはTAITOの**チェイスH.Q**が出るようです。そして，パック・イン・ビデオでは**実戦ビリヤード**を，ザイン・ソフトでは**バルーザの復讐**と**レインフォーザー**をそれぞれ開発中とのこと。では。

POOL BAR

Vessel

ぴくせる君

ああ，ゲーム基板

最近，アーケードゲームの中古基板を楽しむ人が増えている。あの祝一平氏も「ペンゴ」などのオールドゲームをコレクトしはじめたというし，Oh!FMのＹ氏があの「グラディウス３」を買い込んだという話もある。

ところで，ゲーム基板で遊ぶにあたっては，ディスプレイTVとコントロールボックスが必要だ。TVのほうは，X68000のCRT（15kHzが使用可能なモデル）で大丈夫だ。コントロールボックスとは，テーブル筐体の電源部とジョイスティック部をコンパクト化したもので，２万～３万円ほどで買うことができるが，慎重に選ばなくてはいけない。

私はいままで，ゲーセン払い下げのテーブル筐体で楽しんでいたのだが，調子が悪くなったので先日キョーワインターナショナルのKIC-045DX（34,000円）というのに買い換えたのであった。しかし！　なんと１日にして電源がイカれてしまったのである。

ま，通常に使っていて故障したのだから保証書が効くなー，と思って保証書を捜したのだが，KIC-045DXには保証書が付いていないのであった。むむむ！　すぐにメーカーに電話をしたら「それじゃ，今回は特別に無償で修理してあげましょう」とのこと。なにが「特別に」だぁ，この野郎！　NECのパソコンにでさえ１年間の保証書が付いとるぞ。

さて，このKIC-045DX，疑似ステレオ回路なるものが入っているのだが，どうも回路がいい加減らしくて，ヘッドホンをつなぐとゲームによっては画面がフラッシュ（！）したり，まったく映らなかったり（？）というハプニングを起こす。これについてもメーカーに電話で聞いてみたのだが，「うちでもよくわからないんですよ」とのお答え。うーむ，いくらまだ発展途上の業界とはいえ，あまりにも杜撰である。まぁ，なかにはシグマ電子のコントロールボックスARV-7（20,000円）のように，３カ月の保証書を付けているところもあるようだが。ちなみにOh!FMのＹ氏は，ロータスプレスのLAM-1（19,800円）というのを使っているそうである。

読者の皆さんのなかで，このゲーム基板関係の情報があれば，アンケートハガキの隅にでも書いておいてほしいなと願う（善）なのであった。　（善）

THE SOFTOUCH

●三国志II

名作の子はやっぱり名作だったのだ

Urakawa Hiroyuki
浦川 博之

「信長の野望・戦国群雄伝」「維新の嵐」につづいてX1turboに光栄の自信作が登場。あの不朽の名作三国志の続編というといやがうえにも期待は高まってしまいますが，はたしてどのような新しい試みがなされているのでしょう。

X1turbo用 5"2D版3枚組 14,800円(税別)
光栄 ☎044(61)6861

X1turboユーザーはこの際顔をグシャグシャにして喜ぼう。イース3がなんだ，エメラルドドラゴンがなんだ，こちとら光栄という強い味方がいるんでぃ。しかもあの三国志の続編だぁ。

とはいえ，なにせ元祖三国志はいまだにTOP10にハガキがくるほどの名作。あれ以上やりようがあるのだろうかという疑問もなくはなかった。しかしご心配なく。三国志IIは微妙にコンセプトを変えながら，全体には元祖三国志以上に完成度が高かった。だてに4年も待たせてないなと思わせるだけのものがあるのだ。さすがぁ。

特長の巻

三国志IIになってもゲームのおおまかな部分は変わらない。やっぱり土地を開発して税金を稼ぎ，兵をやとってがんがん敵国に攻め込んでいくのだ。そんなことを言うと三国志も信長の野望も同じになってしまうけど，ちょっとコマンドを変えたり，新しいパラメータを追加するだけで全然印象の違うゲームになってしまうのだからあら不思議。たとえば元祖三国志でのコンセプトはいわば「人は城なり」で，まず才能のある人材を得て，その才能に見合った職につけてやるというのが天下への道の第一歩だった。で，三国志IIの特徴は，「人物同士のかけひき」にある。特に外交に関する戦術が充実している。共同作戦を申し込めば，本当に合同して第三国に攻め込んでくれるのだ。ちなみに自分がピンチになったときに援軍を頼むこともできる。

そういうはればれとした外交に対する裏の外交が，新しくできた「計略」というコマンドだ。「埋伏の毒」とか「偽書疑心」というネーミングがいかにもという感じ。「埋伏の毒」は自分の腹心を他国に送り込んで登用させるもので，戦争のときに寝返らせることができる。「偽書疑心」のほうは武将の忠誠度を落とすもの。他国同士を争わせたり，大守に謀反を起こさせたりといった計略がある。こういうあくどい所業がバレると君主の信用度が下がってしまう。信用度が下がるとどうなるかというと，人材の登用や外交がしづらくなって，孤立して，困ってしまうわけだ。情けは人のためならず。計略はバカ正直と言われず，腹黒い奴と言われない程度にしておこう。

国内の巻

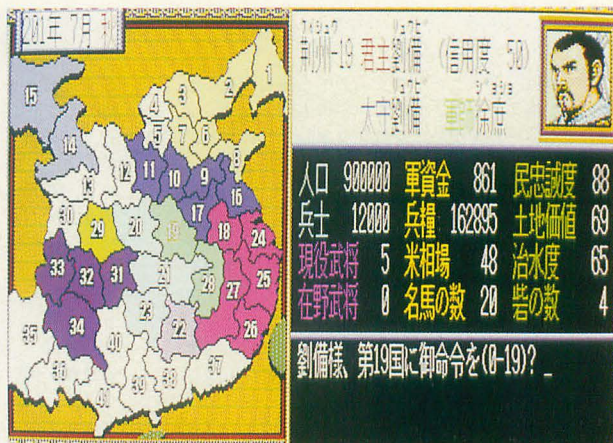
テストプレイにあたって選んだのは，シナリオ3の「劉備荊州に潜み髀肉を嘆ず」。君主は劉備を選んだ。

とりあえず隣の空白地に関羽を送り込むとちょうど写真のようになる。まわりは曹操，孫権，劉表と強豪ばかり。まずどこの領土をもらおうかしばらく考え，孫権のほうに進もうという方針を立てた。曹操と同盟を組んでおけば，いつ孫権に攻め込まれても援軍を頼みに行ける。念のため，劉表とも同盟を組んで，まずは内政の充実だ。

三国志IIでは1国にいる武将1人につき1回命令を出せる。武将の能力を問わないコマンドもあるので，とにかく武将数を揃えることが大切。バカ揃いでもコマンドを出せないよりずっとマシだ。

三国志IIでは自分の作った君主でプレイできるけど，部下がいないからむちゃくちゃ難しい。部下を探そうと他国の様子を見るとその月のターンが終わってしまう。パラメータや年齢によっていろんな顔が出てきてそれはそれで面白いんだけど，初めのうちは手を出さないほうが無難だろう。

税収を増やすにはとにかく土地の価値を上げること。外交に使う武将以外はみんな開発をやらせる。知力が85以上ある武将で金2，それ以下なら3ぐらいが効率がいい。7月と1月の徴収の前には魅力の高い武将に施しをさせるのを忘れないように。人数さえいれば集中的な投資で国の地盤はすぐに整えられる。

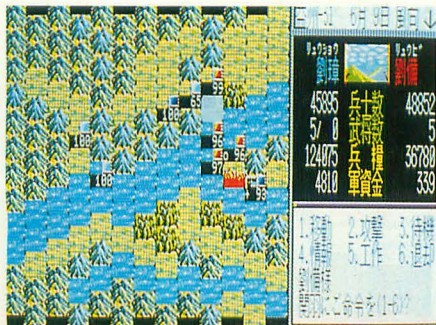
あとは治水に投資して，民の忠誠度に気をつけていればいい。たまに施しをしておけばよかった前作と違って徴兵だ，税の徴収だ，出兵だ，攻め込まれたと，何かあるとすぐに下がってしまう。国力を充実させ

これが基本となる画面。国の数は41に整理されている

るのが簡単なぶん，メンテナンスは大変だ。

自分のターンが終わり，人の戦略を見ていると，ときどき馬のハニワみたいなのが国から国へテケテケ走っていく。これが外交の使者だ。人物を引き抜きにいったり，同盟の申し入れにいったり，あるいは何か計略をしかけようとしているわけだ。ご苦労さんねぇ。で，ときどき自分の領地を通っている使者を捕まえることがある。敵国なら遠慮なく密書を見てしまおう。しかし同盟国の使者にそんなことをすると，相手には嫌われるし，ほかの君主に対しても信用をなくす。ひょっとしたら自分の配下を引き抜きにきているのかもしれないし，部下の武将にスパイがいて「戦時の際は，手筈どおり内応せよ」なんて書いてあるかもしれないから難しい。実際私がプレイしている最中に，同盟国の使者が自分をはさんだ反対側の国へ行こうとしている様子だったので気になって見てみたら，この国を共同して攻め滅ぼそうと書いてあった！

戦争の巻

劉備のいる19国というのは都に近く，戦争もしばらくないので文人がほうぼうから集まってくる。国力ができた頃には徐庶，諸葛亮といった人材を得ているはずだ。正直に戦争にとりかかってもいいが，知力の高い人を軍師に任命できれば計略を使って楽に勝てる。さてどうしようかなと見渡すと，南の22国では劉表の配下だった趙範が謀反を起こして独立している。この趙範が平凡な奴で，部下の管理もいい加減。配下の周倉に寝返ってもらおうと決めて諸葛亮を送ったら，あっさり同意してくれた。これで兵３千はこちらのものだ。わはは。卑怯な策を弄したうえで，22国に１万５千の兵を送り込む。

国内に入るや，張飛が勝手に一騎討ちを申し込んだ。この一騎討ちに勝つと相手の部隊をまるまる捕らえることができるし，むこうが断っても，失望した兵士がわらわらと逃げていく。戦局を左右しかねない大

武将１人につき命令は１回

君主のデータ画面

事なイベントなのだ。趙範側は断ってきた。ふん，臆病者め。

臆病者対卑怯者の戦いが始まった。部隊の移動力は，その訓練度によって決まる。普段から訓練をしておくことが大切だ。次第に近づき，包囲したところで周倉に寝返らせる。趙範，覚悟しろ。

ところが敵もさるもの，そうと知るや火計をかけて部隊を分断させてきた。前作のように１ターンで部隊が全滅するようなことはないし，３日ごとに天気が変わって雨も降るからそれほど困りはしないが，包囲は解かざるをえない。兵の消耗を恐れて攻めあぐねていたら，趙範はさっさとほかの空白地へ逃げてしまった。うーむ，残念。

とにかくこの三国志IIのHEX戦は，見かけこそ前作と同じだが，必ずと言っていいほど計略やイベントがからんできて，１回１回の戦争がドラマティックになっている。私がプレイしたなかでいちばん面白かったのは，孫権に取られた国を取り戻しに攻め込んだときである。孫権側がなんと，こちらが同盟を結んでいたはずの劉表に援軍を乞いにいき，劉表側が２万の大軍をさし向けてきた。ごさーん。「ちっきしょー，裏で手を組んでるなんてきたねーぞぉ」と勝手なセリフを吐いてはみるが，とにかく援軍が到着してしまった以上，こちらにたどり

密書を見ることも大切

つく前にどうにかして本陣を壊滅させなければならない。もう突撃の繰り返しである。ところが，その勢いを見た劉表軍が，今度はこちらに寝返ってしまった。孫権側にとっても，劉表と劉備が同盟を組んでいたのがショックだったらしく，あわてて退却していったのである。あまりの事態の変貌ぶりに，しばしボーゼンとしてしまった。

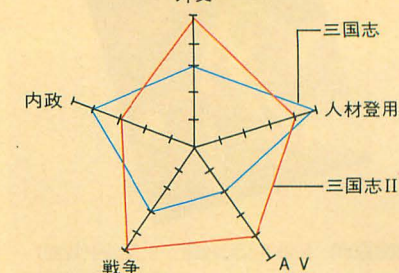
総括の巻

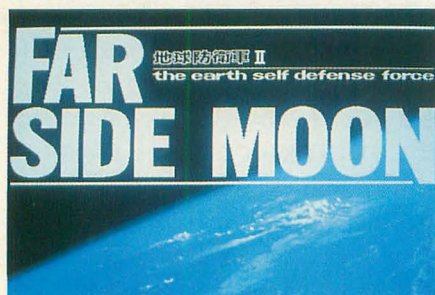
かように三国志IIとはプレイしていて「仮想の史実」を生むゲームなのであった。欲を言えばこちらが同盟相手の敵対国と仲よくしはじめると，同盟国との関係が冷えてくるとか，もうちょっと入り組んだシチュエーションまでサポートしてくれればという気はするが，高望みが過ぎるだろうか。

また，元祖三国志よりも細かい点に改善が進んでいる。武将の選択のときに，関連するパラメータを一緒に表示してくれるようになったし，武将のグラフィックも同じ顔が出てこない。軍師に全般的なアドバイスを乞うことができるのもうれしい。

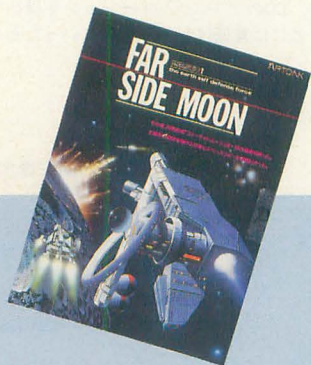
とにかく，あの三国志からまだこんなに新しい楽しみがひきずり出せるとは思わなかった。光栄がゲームを出し続ける限り，X1turboと夜を明かす日々はなくならないようだ。とほほ。

総評

前作に比べて自国の地盤固めに労力を使わないようになり，プレイヤーになるべく他国との交渉・計略をさせるようなゲームデザインになっている。戦争をとってみても，攻撃側の兵の上限が，守備側の半分の10万人で，あとは共同作戦軍か寝返りで補わなければならない。インタフェイス部分はより親切に作られ，ゲームの難易度自体も若干落としてあるようだ。軟派になったという見方もあるが，ゲームクリアまでの過程がより充実したものになっているのだからそれも良しだろう。なお，音楽はカシオペアの向谷実氏ということだが，X1turbo版で聴いてみたぶんには特に何とも思わなかった。FM音源に通じたミュージシャンというわけではないから，CDのアレンジバージョンを聴いてみないことには真価はわからないといったところか。

THE SOFTOUCH

●FAR SIDE MOON―地球防衛軍II

フルメタル感覚のシミュレーション

Kaneko Syunichi
金子 俊一

120年後の地球を舞台にしたシミュレーションゲーム。月の裏側で繰り広げられる侵入者との戦いに勝ち地球の平和を守ること，それがこのゲームの目的だ。さまざまな武器や搭載部品をうまく使いこなせ！

X68000用　5″2HD版3枚組　9,500円(税別)
アートディンク　☎0474(77)7541

地球防衛軍？　どっかで聞いたことのある名前だけど……思い出せない。もちろん，地球防衛軍自体は知っているけど，何か引っ掛かっている。自宅のパソコンの端末を見てみる。生まれてこのかた十数年間使っているデータベースにも載ってないみたいだ。液晶パネルになっている壁に向かっても，幸せそうに寝ているペットの顔を見ていても思い出せない。……えっとぉ？

自己紹介しておこう。僕の名前はザインハルト・ボン・ニャーニャー。ただのパソコン少年である。いつになく暇を持て余していた僕はアルバイトを見つけるディスクマガジン・FROM Cを買ってきたのだ。「世の中ゼニや！」というディレクトリをあけてふと目についた記事がこれ，「地球防衛軍司令長官募集！　高給優遇　週休2日　勤務地ファーサイドムーン」。ファーサイドムーンといえば月の裏側じゃないか。自慢じゃないが僕は地球から一歩も外に出たことはない。修学旅行はアースサイドムーン(月の地球側）だったんだけど，おたふく風邪で寝込んでしまったのだ。今どき(2110年)月にすら行ったことがないなんて特別天然記念物だ。あっそうだ，地球防衛軍といえばずっと前にもプチプチビニールの服を着たおねえさんが司令長官を募集してたんだっけ。あの広告は話題になったもんな。とすると，司令長官ってのは月の裏側でおねえさんと遊んでりゃいいんだ，きっとそうに違いない，これに決めた！

純粋な少年の淡い夢？　を打ち砕き，地球防衛軍の司令長官は忙しかった。それというのも珍しくUE(敵)が攻め込んできたのだ。ちっくしょう，僕の夢を，僕のおねえさんを返せ～！　そうだ，この月面上から敵を追い出しさえすればいいんだ，そうすれば僕の夢はきっと帰ってくるはずだ。おねえさん，待っていてください！　ともかくこれが地球とUEのはてしない戦争への第一歩になったのだった。

長官はつらいよ

僕がしなければならないことは次の5つ。戦闘機を作る，それに載せるソフトを作る，戦闘機を発射する，部品の開発を頼む，部品を補給する。良くいえば宇宙開発事業団，悪くいえばただの雑用係だな。ともかく，その5つの仕事を簡単に紹介してみよう。

1)　戦闘機を作るのはスペースが決まっているので，そこに合わせて部品を置けばいいだけだ。ただし，いくつかの制約がある。エンジンを積まなければならないし，武器は外側に向けないと……(当たり前だ)。まあ，細かいことはやりながら覚えればいいさ。ノリとしては子供の頃よくやった紙飛行機を改良していくようなものだろう。

2)　戦闘機に搭載するCPUがあるわりには意外とソフトのほうは使えない。このCPUはレーダー系に反応があったときだけ攻撃なり回避なり一定の指令ができる。

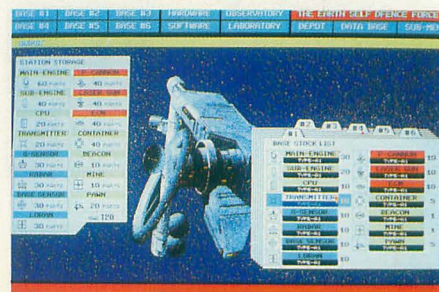
3)　戦闘機の基地は6つあり，合計6機の戦闘機を同時に飛ばせる。運を天にまかせて方向を決めて，「それいけ」とかいいながら発射ボタンを押すだけさ。

4)　部品の開発は地球上で行われるんだ。研究所が6つあるから同時に6つの部品を研究できる。

5)　部品の補給は地球周回軌道上にある軌道ステーションに指示を出す。部品のストック数は決まっているので，へたに飛ばしまくると底をついてしまう場合がある。

こんなことをアルバイトでやってもいいのか？　という疑問を抱きつつも，おねえさんのためならエ～ンヤコ～ラ。

ちょっと解説(ナレーション)

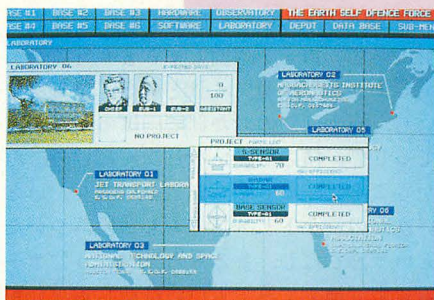
これはシミュレーションゲームです。とはいってもターン方式ではなく，A列車で行こうやポピュラスのようなリアルタイム方式です。ストーリー背景としてはUEが太陽系を侵略しにきたのでプレイヤーは地球を守るために戦う，というもので，目的としては「敵を太陽系から追い出す」ことにあります。そのためには月，火星，木星，土星にある敵の基地を全滅させなければいけません。それにはフルオートで動く無人戦闘機・MACSを使います。戦闘機と呼んではいますが，こいつは搭載するユニットによって爆撃機にも輸送機にもなります。基本は輸送機にロボット部隊を載せ，敵基地にロボット部隊を送り込むことです。基地内ではロボット同士の戦いになり，その基地にいる敵ロボットを全滅させると基地を破壊することになります。月面にある敵基地を全滅させると舞台は火星，木星，土星のように進み，土星の敵基地を全滅させると無事エンディングを迎えることができ

軌道ステーションでは部品の補給ができる

ます。以上,「ファーサイドムーンはこんなゲーム」のコーナーでした。
それでは早速ゲームスタート！

最前線基地ファーサイドムーン

ともかく敵の基地をポーン（ロボットの名前）で破壊すりゃいいんでしょ。いきなりザインハルト長官となった僕はポーン搭載型の輸送機を敵の基地目がけて発射した。それいけ！ やった～，これでおねえさんと遊べるぞ～。喜びもつかの間，敵の戦闘機によって我らが輸送機はあっけなく撃墜されてしまった。やっぱりものごとには順序ってものがあるらしい。まずは，MACSのデータベースをよく見ることにしよう。そこで僕は偵察機ってやつを見つけた。こいつは敵の基地を見つけるとビーコン（発信機）を投下してくれるようだ。幸いポーン搭載型はビーコンのセンサが搭載されているので基地が見つけやすくなるだろう。さらに護衛戦闘機なんてのもあって，味方機を護衛してくれるそうだ。フッこれで勝ったも同然だな，見ていろUE。などといっているうちにUEがベース1にポーンを落としてきた。わ～っ，オレはまだポーンの使い方を知らないんだぞ。こらっ勝手に動くな，敵はあっち！ 自分のビームに当たるんじゃない！ なぜかポーンのビームは反射する。当然，真正面から壁に向かって撃てば痛い目に遭う。壁に向いてるときには撃たなければいいじゃないかとお思いだろうが，甘い。ポーンの向きは勝手にポーンが決めるのだ（しかもかなりちょこまか動く)。これじゃあ自爆も増えるわけだ。「どこに行くんだい？」「ポーンに聞いてくれ」ってやつだね。ともかくポーンにはもっともっと賢くなってもらわなければ。真っ先に研究しなければならない対象だ。大きな教訓を残しベース1は散った。ありゃー，地球崩壊まであと5つか。やっぱり指令長官はアルバイトじゃいけないのか？

＊

人類の英知とアシスタントを総動員して強力なポーンの開発を最優先した。予定では現在の3倍強の強さのものを完成させるのに100日ほどかかるらしい。その間どうやったら残りの基地を守れるのか僕は考えていた。基本的にはUEのポーン搭載型を全部撃墜すればいいわけなのだ。そこで攻撃型のMACSをいっぱい飛ばし，撃墜してくれることを祈ってみた。だけど月はとっても広かった。打ち上げたら何処へ行くかわからないようなMACSがたったの5機ではとてもじゃないが守りきれない，っていうことに気づいたのはUEの戦闘機があと一歩で基地に辿りつくところを見てからだ。ところがどうしてか月面上をMACSが飛んでいる画面の間しか時計が進まない。つまり補給をやっている間などは0秒というわけだ。あわてて補給ステーションを呼び出し，予備のポーンを基地に割り振った。さらに危険が迫っているその基地でコントロールしているMACSが撃墜されてしまっていたので，あわてて攻撃型のMACSを発進させた。それいけ！

そのMACSは発進直後に敵のポーン搭載型の戦闘機とぶつかって爆発してしまった。そっかぁ，そんな手があったんだ……。

＊

僕は世にも恐ろしいMACSを作り始めた。搭載されているものメインエンジン1基，CPU1個という，これ以上シンプルにすると飛ばないくらいに簡単な作り。これはMACSが無人戦闘機なのを利用して，まっすぐしか飛べない最も安上がりな特攻機（ミサイルとも呼ぶ）なのだ。確かに引き付けておけば100発100中，効率もひどいもんじゃない。なんといっても問題なのはポーン搭載型であり，攻撃型なんてのはいくつ飛んでても怖くはないのだ。この作戦は見事に当たった。開発の時間を稼ぐため，ただひたすらによけまくるMACSを1機だけ常に飛ばしておき，あとはミサイル作戦だ。僕って戦略の天才だったのかもしれない。

……ザインハルト少年はすでにベースがひとつ破壊されていることも忘れ，自分に酔っていた。

＊

やがて強力なポーンが完成すると，ポーン搭載型をやまほど繰り出した。今度のやつは馬鹿じゃない(前より)。装甲が厚いし，なんといってもビームの破壊力が段違いだ。動きだけは相変わらずだけど，敵基地まで辿りつけばほっといてもあとはどうにかなる。そんなことを繰り返しているうちに他の部品もパワーアップされ，どうにかこうにか月の平和は守られたのであった。「やったぁ！ これで本当におねえさんと遊べるんだ」。ザインハルトは甘かった。有無を言わさずに火星に連れて行かれたのだ。契約違反だぁ，僕の夢を，僕のおねえさんをかえせ～っ！

＊

こうして太陽系はひとりのアルバイトの少年によって守られたのだ。結局，彼の夢がどうなったかは知るよしもない。

搭載部品研究所で開発計画を練ろう

機体の設計画面だ

戦闘機の戦いっぷりをとくと見よう

最後にひと言

PC-9801では前作の地球防衛軍が人気でしたが，システムとしてまだまだ磨ける部分が残っていると思えます。全体的にプレイヤーが暇なのです。たとえば，常に時間が進んでいて，プレイヤーは敵機を気にしながら開発をしたり，ソフトを作ったりというほうがスリルがあるかもしれません。部品の開発期間が長くて性能が一気に2倍3倍よりも，期間は短いけれど2割増し，3割増しのほうが遊べるのではないでしょうか。さらに完成したのなら研究所から報告があるべきです。補給も輸送機を飛ばしてステーションまで取りに行ったり，他の基地へ運んだほうが(敵に撃墜されるかもしれない)ジレンマ的な面白さがあることでしょう。見たい情報がすぐ見られることも大切です。

とはいえこのゲーム，いい資質を持っていると思えますのでアートディンクさん，続編を作るのならば，がんばってみてください。贅沢をいえばX68000用にはBGMも欲しいし，ディスクを取り換えるときはイジェクトしてほしいな。

項目	点数
グラフィック	8
操作性	4
ミュージック	7
サウンドエフェクト	6
アイデア	7
子供の頃の紙飛行機飛ばし	9

THE SOFTOUCH

●グラナダ

X68000オンリーのアクションゲーム

Kageyama Hiroaki
影山 裕昭

前号でも紹介したこのグラナダですが，開発バージョンであったため商品版とかなり相違点がありました。こんなに違っているんだったら，ぜひもう1度紹介しなくちゃと思い，異例の2度レビューを行うこととあいなりました。読者の皆さん，もう1度お付き合いください。

X68000用　5"2HD版4枚組　8,800円(税別)
ウルフ・チーム　☎03(5273)4795

皆さんゲームやってますか。どうも，影山です。あと2週間もすると6月になってしまうんですね。6月というと祭日はないし，梅雨前線のおかげで雨ばっかり降ってるしで，僕の大嫌いな月なんだけど，だからこそゲームで楽しい気分になりたいというもの。テトリスブームも一段落ついたという感じだし，そろそろアクションゲームに乗り換えてみてもいい時期じゃないかな？　というわけで，グラナダです。

グラナダのゲームシステム

ゲームの目的はタンクを操って，マップのあちこちに点在するクリアポイントを破壊すること。画面右下のレーダーに自機とクリアポイントの位置が常時表示されているから，それを頼りにゲームを進めることになります。すべてのクリアポイントを破壊するとその面の守護人である巨大ボスが出てくるから，そいつを倒せば1ステージクリア。タンクを操りマップ上を東奔西走し，レーダーを頼りにクリアポイントを破壊していくというゲームシステムは，「サンダーフォース2のエクセリオンが地上を走っているのと同じじゃない」といってしまえばそれまでだけど，あくまでこのゲームはサンダーフォース2とは似て非なるものです。ほかにもいろいろと参考にしているゲームはあるようだけど，それらをまとめてまったく別のゲームに仕上げているところがスゴスバなんてす。

このテのアクションゲームはとかく単調になりがちですが，それを解決すべく5種類のオプションを用意してゲーム性を高めていることにも注目です。またグラフィックも美しく，ステージごとに特色があって一見の価値ありという感じ。操作方法も工夫されていて，Bボタンを押しながら移動キーを押すと砲塔の向きを固定したまま移動できるから，前方を攻撃しながら後退するということも簡単だし，A,Bボタンを同時に押せば強力な波動砲（ブラスターというそうだ）を発射できます。タンクの移動速度も地形や障害物によって変化させるなど，細かいところにまで気が配られています。さらにアクションゲームは大の苦手，という人でも難易度がEASYからMANIAまで自由に変更できますから，自分のレベルに合わせて遊べます。こういった操作性や細かい設定，BGMもMT-32に対応させるなど，ゲームを面白く遊ぶための配慮が随所に見られます。

さて，いいことばっかり書いてきたけど，このゲームには僕の嫌いな時間制限があるんです。時間制限はゲームに適度な緊張感を与えてくれるから悪いものではないけれど，僕は個人的にこの時間制限が嫌いなのだ。これも自由に設定できるようになっていたらうれしかったのに残念だなあ。

システムの説明は，まっこんなところかなっ，という感じ。で，僕はこのゲームの魅力はボスキャラとの対決だと思うので，やる気をソソルためにも，あえて後半ステージのボスキャラの写真は掲載しません。ぜひ皆さん自身の実力で確かめてください。

ステージ1

ステージ1は市街地戦の雰囲気をかもし出している。クリアポイントの数は18個とたくさんあるように感じるけど，1カ所に集中しているから思ったほど手間はかからないはず。マップの左下から時計回りの順番でクリアポイントを破壊していこう。

マップ上方と中央右側あたりにはローラーがゴロゴロと転がっていて，行く手を阻んでいる。これにひかれるとさすがに受けるダメージは大きいから要注意。ローラーとローラーの間の矢印の部分にブラスターを2，3発お見舞いすれば破壊できるぞ。

ここではマップ右下のクリアポイントを最後に破壊するように残すのが定石。頑張ってそこまで辿り着いたら，残った6個のクリアポイントを破壊。すると，お待たせしましたのボスキャラの登場である。こいつはピョンピョンとあちこちを飛び回るから，下敷きにならないように避けて，着地している4本の足を狙って攻撃しよう。

ステージ2

舞台は一転して上空へ。眼下にはいくつもの雲が現れては消え去り，見た目にも非常に美しいステージ。クリアポイントは輸送機の8機のエンジン。ひねくれ者の僕は，こんなところにどうやったらタンクが登れるんだろうと考え込んでしまった（どうせ僕は暇なヤツですよ）。

スタートしたら前進していくとオプショ

こいつがステージ1のボス

ンが落ちているから，まずはそれを装備しておきたい。同じオプションが左右両翼端にもあるから取り逃さないように，ちょっと考えればわかるが，上空ってことは，足を踏み外せばしゅぅぅぅぅと地上めがけて墜落していってしまうってこと。注意が必要だ。そうそう，穴に落ちても結果は同じだから試そうなんて思わないでよ。

このステージのボスは，暗闇からぐるんぐるんと回転しながら突然と姿を現す。なかなかカッコいい演出だ。1面のボスと比べると火炎を飛ばしてきたりしていくぶん手強く，苦戦を強いられることになるかもしれない。こいつに関してはむやみやたらに攻撃していると，敵の弱点がシールドに覆われてしまってなかなか攻撃できない。コツとしては砲塔を弱点に向けたまましばらく攻撃の手を休めると，シールドが開いて弱点があらわになるから，馬鹿なやつと思いつつ，すかさず攻撃してあげよう。

ステージ3

ここで登場するクリアポイントは，いままでのように固定されているものではなく，ちょこまかと動き回っているからブラスターの狙いがつけづらく難しい。ボスは最初に左右のミサイルポッドを破壊することに集中しよう。後ろに回ればそこは安全地帯だ。こいつを破壊すると中央部がパカッと開いて，いきなり火炎放射を浴びせてくる。2面のボスとは比べものにならないくらい強烈。ここはひたすら火炎を避けて，粘って粘って攻撃するしかないようだ。タイムオーバーがあるから，素早く正確な攻撃能力が要求される。

ステージ4

イースの洞窟を思い起こさせるが，明るい部分以外の暗がりでも敵の姿を確認できるから，不意な攻撃で玉砕ということはまずない。クリアポイントは戦車だから，動き回るうえにやたらと攻撃してくるので，なかなか手強い。このステージはオプショ

ステージ2は輸送機が舞台

ステージ3のボスはこんなやつ

ンがたくさんあるから，できるだけ多く装備して攻撃力を強化しておきたいところだ。初めてプレイするときはクリアポイントになかなか辿り着けないと思うが，根性を出して頑張ってほしい。

ボスがなかなか見つからない人，そんなときは心を無にして瞑想に入れば，ボスの出現位置が見えてくる……おいおい，信じないでよ。そんなんじゃゲームオーバーになっちゃう。マップ左上あたりが怪しいぞ。

ステージ5

このステージもグラフィックが美しい。特に海面のように見える部分がキラキラと輝いているようにも見えて，筆舌に尽くしがたい美しさだ。ちなみにこの部分を移動するとポチャンとはまってしまいそうな錯覚を起こすが，タンクはちゃんと移動できるから安心してほしい。敵もだいぶ手強くオプションの役割は重要だ。僕が遊んでみた限りでは，3カ所にオプションがあるから是非モンで全部装備しておきたい。

ボスは手強い。ひたすら避けて，避けて，オプション&ブラスターの1－2アタックで切り抜けるしかない。

ステージ6

ここにはクリアポイントが初めから存在

オプションをうまく利用すべし

しない。目指すはボス，それだけだ。ボスに辿り着くまでワンサと戦車が出てくるが，いくつもの困難を乗り越えてきた君の腕なら，目をつぶってでも切り抜けられることを期待している。僕なんか足でジョイスティックを操作してますから。ワッハッハ。

さて，途中にあるゲートをブラスターを使って破壊したら，中に入って左と右にひとつずつあるオプションを装備，そして進撃するのみだ。ボスについてひと言いわせてもらえば，「ムーンクレスタ」みたい。イジメだぞ！　この難しさ。残るはステージ7，8。ここから先は見てのお楽しみ。頑張ってね。

*

ゲームミュージックが重要な位置を占めるようになってきている今日のゲーム業界のなかで，僕はグラナダの効果音の出来のよさに感心した。特にクリアポイントを破壊したときの爆発音が好きで，これを聴きたいがためにゲームを立ち上げることもしばしば。また，ウルフ・チームが大好きなビジュアルシーンに台詞がついていないことにも意外性が感じられた。あの口パクが見られないのは残念だったけど，これも何か狙うところがあってのことでしょう。

とにかくこれだけは断言できる。

グラナダは純粋アクションゲームだ！

総　論

アクションゲームは，一度クリアしてしまえば，もはやそのゲームに対する魅力はグーンと半減してしまうのが普通である。ましてエンディングが CONGRATURATION! と表示されるだけだったりすると，なんじゃこりゃ！　である。

X68000のアクションゲームを背負ってきたサンダーフォース2やジェノサイドは難易度が高く「どうだ難しいだろう，クリアできないだろう」といった，見方を変えれば素人プレイヤーに対する挑発的な態度をとっていたわけで，それに対して僕なんかは「クリアしたるわい」と夢中になっていた気もしないわけではない。

しかし，グラナダは難易度が自分の実力に合わせて選べるし，ましてやEASYモードでプレイしたりすればさして難しいゲームとはいえず，先陣のソフト群のように，とうとうクリアできずに途中でさじを投げてしまう人は少ないだろう。もちろん，サウンド，グラフィックなど，平均以上のレベルである。

誰でもできるアクションゲームはこれが最初かもしれない。あえてグラナダを軟派とするなら，先例は硬派ということになるか？　結局はアクションゲームの捉え方の違いなのだが，これに関しては意見の分かれるところだろう。

項目	点数
サウンド	7
グラフィック	9
操作性	8
ビジュアルシーン	8
スコア	10
熱中度	8

THE SOFTOUCH

●ワンダラーズ・フロム・イース

マドルのイースⅢ冒険記〈後編〉

Nishikawa Zenji
西川 善司

3カ月にわたってお送りしたマドルのイースⅢ冒険記もいよいよ最後。あなたのアドル君はどこまでいったでしょうか。えっ，もう終わっちゃった？ うーん，そう言わずに……。

X68000用 5"2HD版4枚組 8,700円(税別)
日本ファルコム ☎0425(27)6501

ドギ「彫像は4つ，すべて揃ったようだな」
マドル「そうだな。アイテムウィンドウのSTATUE（彫像）という枠も4つ分しかないしな」
ドギ「…そういう身もフタもないようなことを言うなよ」

俺様はマドル＝リステリン（うがい薬じゃないぞ），あのイースシリーズのアドル＝クリスティンの従兄弟（いとこって読むんだぜ，知ってたか。俺はいま知った）の親戚の近所に住んでいる。ドギが師匠と慕う人物の協力により，エルダーム山脈に眠っていた4つ目の彫像を手に入れることができた。俺たち一行は足取りも軽く帰路についていたのだった。

ドギ「マドル……。町の様子がおかしいと思わないか？」

そして再び使いっぱ

ドギ「ガードナーさん，どうしたんだい？何かあったのかい」
ガードナー「大変なことになった。バレスタイン城のやつらが現れてこの街の人間を人質として連れて行ってしまったのだ」
ドギ「マドル!! お前ならやれる。バレスタイン城へみんなを助けに行くんだ」
マドル「……」
ドギ「…どうした？ マドル」
マドル「…いや，イースシリーズって本当に主人公が使いっぱとしてこきつかわれてないか？」
ドギ「…師匠からもらったヒールリングをやろう」
マドル「こいつ……！」

結局バレスタイン城に行くことになってしまった。出かける前に装備をチェックしよう。バレスタイン城はとんでもなく広いところだぞ。よって，できることなら1回も町へ戻らずに先へ進みたい。バンデッド・アーマーに，バンデッド・シールド，両方買い揃えたいところだ。さて，全国のアドル君たちは，いま，アイテム類は何を持っているのかな。え，薬草しか持ってないって？ 幻影の鏡にアミュレット，いままではあまり使う必要がなかったかもしれないが，ぜひ買っておこう。そうそう，お店のお姉さんは君にもう「プロシアの秘薬」を売ってくれるかな。これは，消費したリングパワーをある程度回復してくれる大変便利な薬なんだが……。え，売ってくれない？ さ，さては君は先月のOh!Xを読んでないなぁ！ ティグレー採石場にある，あるアイテムを取り忘れているぞ，すぐ取りに戻ろう。

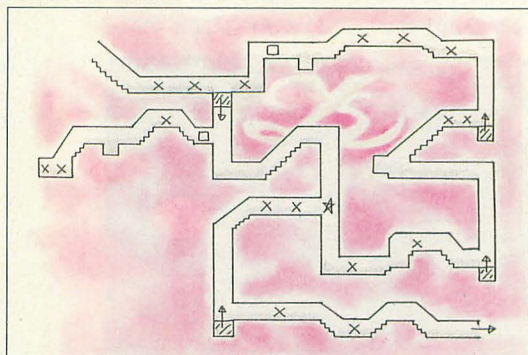
バレスタイン城へ

ここは，バレスタイン城の前だ，おぉ！城の後ろを流れる夕焼け雲が美しい。おや，あそこに立っているのは？

マドル「エ，エレナ，なぜ君がこんなところに……」
エレナ「レドモンドの街があの城の人たちに襲われたでしょう。ここへ来たら兄さんに会えるんじゃないかと思って……」
マドル「エレナ，君は……，えれーなー(偉いな)。なんちって，ぱっくん」
エレナ「……」

名残り惜しいエレナの冷たい視線を振り切ってやって来たのは，バレスタイン城地下1階。ここでは蟬（セミと読むんだぜ。あぁ，今日君は2つも漢字を覚えちまった，よっこの勉強家！）の幼虫のようなザコキャラが登場する。こいつは一定距離以上近づくと猛スピードで突進してくるおしゃま（死語）なやつだが，まともに相手にしていては身がもたない。タイミングよくジャンプしてかわすか，幻影の鏡で敵の動きを止めてからサクサクゆっくりとやっつけるかしよう。前にも言ったが基本的に冒険の「行き」にはダメージを食らってはいけない。よって厄介なザコキャラに出くわしたらこの幻影の鏡やアミュレットをうまく使って切り抜けよう。そうそう，イースⅢではレベルアップすると体力の上限が増えるのはもちろんのこと，ダメージまで回復してくれちゃうので，あともう少しでレベルアップ，というときには多少のダメージは気にせずガンガン先へ進もう。

城にはイシュゲスト君とハルヴァイガー君の2種類のザコ敵キャラが主に登場してくるが，こいつらは毎度言っている，剣を下突きの状態にしてジャンプして敵を飛び

ガルバラン島迷宮

×は敵キャラ出現場所
□は意味不明の紋章のある位置

越しながら攻撃する方法を使ってやっつけよう。ハルヴァイガー君は一定距離以上近づくと急に攻撃速度を上げてくるぞ。意表を突かれるなよ。イヒョー，なんちって。

城の最上階の黒い騎士モーニングスターは，はっきりいって弱い。決して薬草なんか使わないように，やくそうく（約束）よ。なんちって。ぽっくん。

犬がいぬ

ほー。ここが中庭だな。なかなか綺麗なところじゃないか。うわ，なんだこの犬は。放し飼いにしているのかここの城主は。こいつ，放せ。どわーっあちちちっ。火を吐きやがった。しゃがめば火は当たらないだろ，ほふく前進，ほふく前進……と。あちちち。頭が焦げてしまった。どうやら，タイミングを見計らって切りつけ，相手が火を吐く体勢になったらすかさずその火の届かないところへ逃げるのが賢明なようだ。あら，なんだ，あっという間にやっつけられちゃった。アドル君達，ここも薬草は使わないで倒そう。やくそうく（約束）よ（しつこいな俺も）。

先は長いぞ。がんばれ。

ガルバラン島

エレナをさらった，にっくきガルバランを倒すためイースIII最後の舞台であるガルバラン島に俺は足を踏み入れた。装備はすでに最強になっているかな。フレーム・ソード，バトル・アーマーにバトル・シールド。持っていなかったらバレスタイン城へ取りに戻ろう。

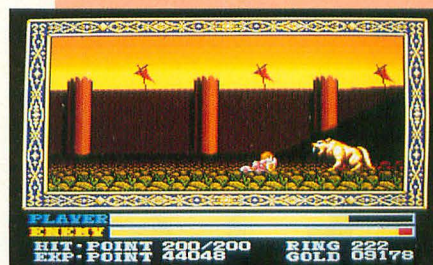
ガルバラン島の洞窟は真っ暗闇だ。出発前にエドガーさんからもらった「鬼火の宝珠」で明かりをと……。なんだこりゃ，足元もろくに見えやしないじゃないか。うわっと，いきなりガイコツのスケルトン君。あ，弱いネ，君。あっという間に倒せちゃった。おっと，床が下に動き出した。どうやらエレベータらしい。えーい。道なりに進んじゃえ。…あ，どうも同じところをぐるぐる回っている気がするな。しかたがない。マップでも描くか(図1)。マップ中の☆印の部分で左の通路に着地できないと同じコースをぐるぐる回る羽目になるぞ，注意されたし。

暗闇を抜けると……。がら－んとしたところに出てきたな。もしかして……。やっぱり中ボス・ガーランド君でした。バレスタイン城でさんざんな目に遭わせたのに，また懲りずに出てきたね，君は。多重スクロールする柱の陰に隠れたりするので少々てこずるかもしれないが，倒せない相手じゃないだろう。しかし！　しかしだ。こいつはいままでのボスとは違って，倒したあとの体力回復サービスがないのだ！　そうそう，しかも一度ガルバラン島を出ると復活してしまう。つまり，アイテム類を買い出しに町へ戻ったり，体力を回復するために島の外に出たりすると倒したはずのガーランド君がまた出てきてしまうのだ。だから，薬草を使うなんてのはもってのほか，(最後のボス，ガルバランに備えて）ほとんどノーダメージで倒して，先へ進まなければいけないのだ。そうだ，それを考えるとブロシアの秘薬（3回使えるリングパワー回復剤）なんかはまだ一度も使っていない新品のを持ってくるべきだったな。

ガルバラン君ガンバル

おっと，行数が残り少なくなってきた。本ボス・ガルバラン君は，いままでのボスとは比べものにならないほどに強い。あきらめてここでやめてしまっている非アクション派のアドル君もいると聞いている。

まず，はじめにガルバラン君の両腕を切り落とさなくてはならない。パワーリングを身に付け，手から放たれるエネルギー弾をよけつつジャンプしながら剣を振って手を攻撃しよう。ちと，せこいが，リングパワーを節約するため攻撃不可能な位置にガルバランがいるときにはパワーリングは，外しておこう。一定時間エネルギー弾を放ったあとガルバラン君は下へ突撃してくるが，決してこのとき潰されてしまわないように。逆に，この突撃してくる瞬間を予測して，下に降りてきたときに一気にダメージを与えよう。腕を落とす前に薬草を使ってしまったり，リングパワーが150以下になっているようじゃまだまだ練習が足りないぜ。

腕のなくなったガルバラン君は，大胆にも急所（心臓みたいなヤツ）をチラリチラリとちらつかせるようになり，しかも突撃行為をしなくなる。だが今度は左右に分裂する波動砲みたいなのを撃ってくるぞ。これは伏せるか，ジャンプして避けよう。さて，急所への攻撃方法だが，これはタイミングよくガルバラン君の懐に潜り込みパワーリングを装備して一気に斬りつけるのを基本形とする。この懐に潜り込むという行為がなかなか難しいのだが，タイミングを見計らって待っている間は，パワーリングはリングパワーがもったいないから外しておこう。ヒールリングなどで体力を少し回復しておくのもいいぞ。また，リングパワーが残り少なくなったら，ためらわずブロシアの秘薬で回復させること。あと，これだけは言っておこう，「プロテクトリング」は役に立たないから使うな，リングパワーがもったいない！　技術よりも気力だ。ガンバレ！

最後に読者へ質問。ミス・リリア・コンテストの優勝者って，いったい誰になったの？　誰か教えて！

言ってくれるよなぁ，ホントに

犬とこんなふうに向き合うハメになるとは……

移植にあたっての改善点

PC-88版からX68000版への移植にあたって，いくつかめざましい発展を遂げた点がある。まず，1つ目はディスクキャッシュが搭載されたということ。アイテムウィンドウのオープンやショップの出入りなどのディスクアクセスは1度行うと2度目は行わないので，イライラせずにスピーディにゲームが進められる。2つ目はディスクのロード/セーブに関して。PC-88版では20秒くらいかかっていたのに対してX68000版はほとんど瞬間で終了する。しかもセーブは自動的に「レドモンドの街」のようなファイルネームが付けられるなど，かなり親切設計になっている。3つ目は動きがかなりスムーズになりアクション性がかなり増えたこと。この辺はX68000ユーザー向けにアレンジしたと考えられなくもないが，やはり初めてX68000でゲームを作っちゃうとこういうふうになっちゃうんだろうな（少し自慢）。最後はBGM。原曲はFM3声＋PSG3声で鳴っていたのがX68000では一気にFM8声＋AD PCM1声へパワーアープ。ゲームミュージックファンはぜひチェックしておきたい1作となっている。

項目	評価
グラフィック	7
動き	8
サウンド	7
難易度	8
操作性	10
ユーザーに対する気配り度	9

THE SOFTOUCH

AFTER REVIEW

AFTER REVIEW

ゲームにどっぷり浸れたゴールデンウィークも過ぎ去り，そろそろ梅雨の季節ですね。うっとおしい雨だからこそゲームでうっぷんを晴らしましょう（とかなんとかいって，年中ゲームを奨励している気がする……)。

ダンジョンマスター

▶テーブルトーク並みの自由な行動パターン。加えて，リアルタイムなアクション性。これはいい。　広島県・実森　仁志（18）

▶いまや，身も心もダンジョンマスターでございます。けど，たまにはお買い物とかもしたい。　京都府・谷口　生美（26）

▶無条件にチャンピオン!!　大阪府・陣山　達夫（20）

▶いままでにない，ゲームの楽しさを教えてくれた。　北海道・島瀬　真基（18）

▶とにかく飽きがこない。次のトラップが楽しみだ。　奈良県・川脇　泰宏（16）

▶ダンジョンマスターには音楽がない。どこまでもリアルで，ある夜ひとりで明かりを消してPLAYすると「ア～！」。私は夜道をひとりで歩けなくなった……。でもダンジョンマスターは私の知るRPGのなかでも最高。　茨城県・菊田　俊彦（31）

▶ドラゴンクエストとはまったく異なるRPGでありながらも，ドラクエを超えたと思っている。　静岡県・富永　恵隆（19）

▶よどみなく流れる時間と尽きることのないイベント。　愛知県・本間　晃（19）

▶やはりアメリカ生まれのゲームは面白いのでしょう。　福島県・樋口　雅人（17）

▶後半は背中がぞくぞくするのがいい。　石川県・紺谷　健（20）

このゲームは，読者の皆さんにもとても好評のようですね。今月のTOP10でもご承知のとおり，といったところでしょうか。このゲームが好評の理由のひとつは，BGMがない点かもしれません。本当のダンジョンの中では，音楽なんて鳴っているわけはないし，だからこそ敵と戦ったり，物を投げたり，穴に落ちたり，そんなときの効果音がよりリアルに感じられるのではないでしょうか。また，キャラクター自体の行動もわりと自由度があるので，自分自身がダンジョンにはまっている感覚でゲームを進められるからかもしれません。

それに，たとえロード時間が長いとはいえ，たったディスク2枚で何カ月も遊べるRPGなんて，そうあるもんじゃありません。ディスクの枚数が少なくても，いいゲームはできるということの証明ですね。

まあ，当分の間このゲームの人気は落ちそうもありませんね。まだやってない方にも十分おススメできます。

X68000用　5"2HD版2枚組　9,800円
ビクター音楽産業　☎03(423)7901

発売中のソフト

★ファーサイドムーン

異変は月の裏側で起こった。侵略者U.E.の突然の襲撃から，太陽系をめぐる戦いはその幕を開けたのである。

いつも独特のシステムでゲーマーの高い評価を得ているアートディンクから，地球防衛軍の第2弾が登場。再びプレイヤーはU.E.に立ち向かう戦闘機を設計して戦場に送り込む。今度は基地の爆撃任務も加わって，いっそう戦略的な思考を要求される。パーツの開発，ソフトウェアの設計と山積みされた課題をクリアして，太陽系に平和を取り戻すのだ。

X68000用　5"2HD版3枚組　9,500円
アートディンク　☎0474(77)7541

★びんびん麻雀ピーチエンジェル

「びんびん麻雀」はイロモノがお好きな方のための麻雀ソフトだ。「お助け麻雀」モードでは，捕らわれた女の子を助け出すために6人の女性雀士に挑む。ほかのメンバーも交えた4人麻雀で半荘4回のトップを取れば，助け出された女の子がお礼にムフフというわけ。6人の敵を倒すとナイショの「スペシャルステージ」に突入だ。普通に麻雀を楽しみたい向きにはフリー対戦モードもあるぞ。

X68000用　5"2HD版　4,900円
ブラザー工業　☎052(824)2493

★維新の嵐

すでにX1turboシリーズには発売されていた「維新の嵐」がX68000にも移植完成。シミュレーションには定評のある光栄の意欲作だ。この維新の嵐では国取りよりも，思想の統一がメイン。さまざまな意見を持つ幕末の英雄を相手に，日本の行く末を論じて相手を説得していく。日本の明日を決めるのは君だ。

X68000用　5"2HD版2枚組　9,800円
光栄　☎044(61)6861

新作情報

★ルーンワース～黒衣の貴公子～

T&Eの放つ新作アクションRPG。物語は山賊の頭領の息子である主人公が「秘風剣サータルスを取ってこい」と命じられたところから始まる。行く先々で現れる仮面の男の正体は，そして主人公の出生の秘密とは。これらのストーリーを一本道ではなく，プレイヤーの行動に合わせて進めていくのが特徴だ。あのハイドライドを生んだT&Eの自

ポピュラス

▶このゲーム，自然破壊ゲームですね。山は削るわ，埋め立てられるわ。

京都府・福知　健（19）

▶神が地球に増えることは，平和のためによい！　と思う……。　大阪府・遠藤　勇（33）

▶もう十分日本でも有名となった，あちらもののポピュラスやシムシティ。BBSで話を聞くたびにやりたいと思ったものよ。

神奈川県・中内　崇夫（21）

▶とにかく面白い。特に地震を起こしたりすると楽しい。　東京都・福江　高志（22）

▶無限に遊べる点。操作性のよさ。アイデアの勝利。他人にぶつけられない残忍さをこのゲームで出してカタルシスを味わえる。

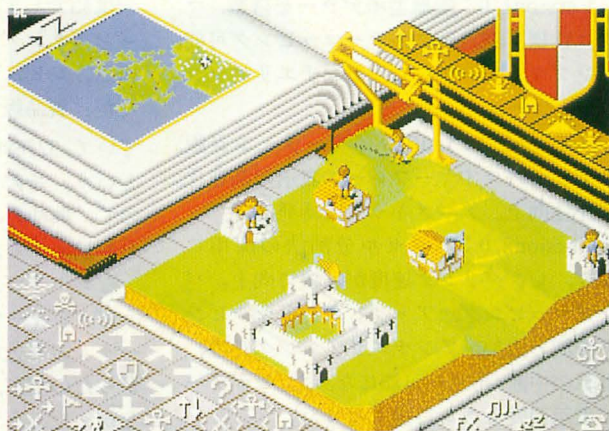
大阪府・纐川　富久（18）

▶宗教的なものが好きだから。

栃木県・福田　正広（21）

▶なかなか奥が深いゲームだと最近思っている！

千葉県・根市　浩（26）

▶ポピュラスを友人とワイワイとやっているとき，友人Aが騎士を殺そうと地震を起こした。が，騎士はしぶとく生き残り，残虐を極めていた。そこで怒った彼は，周りの土地をすべて削り海にしたのである。そして騎士は海のもくずとなったのである。その瞬間からポピュラスは「平安京エイリアン」になった。

東京都・加藤　信之（20）

▶人間の本質がわかる。

福岡県・嶋　嘉和（21）

▶対戦モードで友達と遊ぶもよし，全面制覇を目指して日夜マウスをクリックし続けるもよし，麻薬のような習慣性を持つゲーム。ルールは単純明快，それでいてマンネリ化せず，地形や相手のレベルを考えていくという戦略的要素も失っていないバランスのよさがいい。まだ一度もやったことがない人も怖くなんかないからさ，一緒にはまろうよ。　（純）

このゲームも人気が高いですね。さすがTOP10の2位だけあります。やはりこのリアルタイム性が意欲をそそるからでしょうか。地震や洪水を起こしたり，思わずマウスを握る手も汗ばむというものです。それにしても，神様というのは博愛主義者じゃないんでしょうかねぇ。

X68000用　5"2HD版　9,800円
イマジニア　☎03(343)8911

ワンダラーズ・フロム・イース

▶ドラクエⅣ以来，ひさびさに夜の9時から次の日の朝7時までプレイしたゲームだった。

徳島県・森上　晶仁（17）

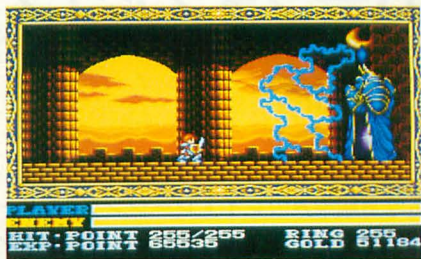
▶YsⅢは，ゲームランクによってエンディング音楽が違うんですね。そこで，EASYで最後のボスキャラを倒したあと，NORMALorHARDのディスクにセーブし再びロードすると，苦労せずにそれぞれの音楽を楽しめます。

埼玉県・大貫　正幸（21）

▶アクション性が強くなったものの，昔のYsの雰囲気がところどころに見られてうれしいかぎりです。特にアイテムを見つけたときの音楽なんかが……。やはり完成度はピカ1。

神奈川県・臼渕　啓明（24）

Ysがいよいよ X68000にも登場というわけで，心待ちにしていた方も多かったはず。その期待に見事に応えてくれる出来だったことはいうまでもありませんよね。シナリオ，グラフィック，どれをとっても安定しているという感じがします。

X68000用　5"2HD版4枚組　8,700円
日本ファルコム　☎0425(27)6501

信作というから要注目。

X68000用　5"2HD版　価格未定
T&Eソフト　☎052(773)7770

★パズニック

TAITOの秀作パズルゲームが，ブロダーバンドジャパンから発売される。フィールド内に積まれたブロックは，それぞれ同種類のブロックと隣接すると消滅する。この性質と重力をうまく使ってブロックを動かし，制限時間内にすべてのブロックを消滅させるのだ。全面クリアには素早い読みと思考能力，そしてタイミングを計る運動神経が必要だぞ。

X68000用　5"2HD版　7,800円
ブロダーバンドジャパン　☎03(341)1135

★アソコの幸福

「ねじ式」でアドベンチャー界に新風を吹き込んだ名作浪曼文庫の第2弾は，ひさうちみちお原作の「アソコの幸福」だ。テーマは日常生活に潜む変態性，そのテーマに鋭く迫る。「永遠に続く偏見」「変化のない毎日」「ヌルヌルさん」「電動コケシくん」といったキーワードから，あなたは何を受け取るか。すべてはプレイしてみてからだ。

X68000用　5"2HD版　価格未定
ツァイト　☎03(299)0461

★天九牌・桃源の宴

中国のちょっと変わったテーブルゲーム天九牌。ハッタリをかましたり，グルになって他人を破産に追い込んだりといろいろ楽しめるけど，その楽しみをさらに広げてくれるのがこの「桃源の宴」。女子高生から美人教師まで，美少女6人があなたのお相手を務めます。勝てばもちろんうれしいグラフィックが……。

X68000用　5"2HD版　3,600円
ブラザー工業　☎052(824)2493

★チェイスH.Q

さぁ，ひさびさのX68000用カーレースゲームの登場だ。TAITO（おお，またか）の「CHASE H.Q」を日本デクスタが移植。ターボチャージャー装備のパトカーが，犯罪者の車を追って時速300kmで駆け抜ける。犯人の車に体当りする過激なカーチェイスは迫力満点だぞ。

X68000用　5"2HD版　価格未定
日本デクスタ　☎03(839)4711

★ワールドコート

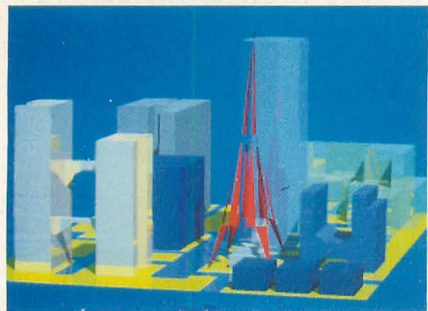
ゲーセンやPCエンジンでお馴染みのナムコのテニスゲーム。慣れてくるにつれてスマッシュやボレーなどの技もできるようになる。友達とやればより白熱するぞ。

X68000用　5"2HD版　8,800円
SPS　☎0245(45)5777

●お詫びと訂正

4月号の「人気ゲームのシステムを探る」の中に誤りがありました。銀河英雄伝説に関する記事中に「音楽が標準システムで出なかった」とありましたが，実際には本体のみでBGMを楽しむことができます。お詫びして訂正します。

THE SOFTOUCH

●サイクロンExpressα

人気CGツールがバージョンアップ

Tan Akihiko
丹　明彦

もうお馴染みのレイトレーシングCGツール「サイクロン」が，またまたバージョンアップして登場。今度のサイクロンは，Z's TRIPHONY DIGITAL CRAFTのポリゴンデータを使用できるようにもなった。

X68000用　5"2HD版　98,000円(税別)
アンス・コンサルタンツ　☎092(522)6347
(東京)☎03(447)4144

久々にサイクロンがバージョンアップした。サイクロンとはデビュー以来の付き合いであるが，最後にレビューしたのはもう1年近く昔のことだ。ちょっと振り返ってみよう。

Version1.0　1989年2月号
Version1.2　1989年5月号
Version2.0 Express　1989年8月号

読み返してみたのだが，我ながらいい加減である。サイクロンを買わなかった人には，なんのことやらわからないことばかり書いてあるではないか。興味のある方はバックナンバーを読み返していただけると，サイクロンの成長のあとを見ることができるだろう。きっと。

サイクロンは非常に不満の多い(失礼！)システムとしてスタートしたが，バージョンアップを重ねるごとに使いやすくなってきている。その足跡は……，

Version1.0：階層型モデラーを搭載，マクロ仕様可能。テクスチャマッピング可能。バンプマッピングなし，アンチエリアシングなし。

Version1.2：1.0のマイナーチェンジ版。モデラーのワイヤーフレーム表示の高速化。操作性向上。アニメキット同時発売。

Version2.0：ボクセル分割法の採用により，レンダリング速度が大幅に向上。バンプマッピング，アトリビュートマッピング機能の追加。アンチエリアシング処理。計算の中断/再開が可能になった。モデラーの表示速度もさらに速くなった。

そして，今回のサイクロンExpressαの登場を見ることになる。「α」はプラス・アルファのαだということだ。なにがプラス・アルファなのか。そう，今回はなんと，

ポリゴンのレンダリングが可能になったのである。

これまでレイトレーシングといえば，金属やガラスの球というのが通り相場であった。要するに2次曲面を主とするプリミティブの組み合わせでシーンを表現するものがほとんどだったのである。これはもちろん，2次方程式を解くのが簡単なため，インプリメントがしやすいからである。

ところでポリゴン(polygon)とは多角形のこと。つまり平面なのだが，多角形はただの平面ではない。頂点を結ぶ線分，つまり辺で囲まれた内側の領域である。だから，ただの平面より情報が多く，構造も複雑になる。ちょっと考えると2次曲面のほうが複雑な物体のように思えるが，ことレイトレーシングに関しては，ポリゴンは通常のプリミティブよりはるかに処理が厄介である。これまでのサイクロン(というよりもパーソナルコンピュータ用のレイトレーシングのほとんどすべて)には，プリミティブのひとつとして，「平面」はあったが，それは無限に広い平面。3角形などを作ろうと思ったら論理演算をうまく使わなくてはならず，それはひたすら面倒くさい。そんな凝った表現にはほとんどお目にかかったことがない。ところが，今回はプリミティブにそのポリゴンが加わったのである。

では，モデラーが大幅に拡張されたかというとそんなことはない。実は，

Z'sTRIPHONY DIGITAL CRAFTのポリゴンデータをもらってきてレンダリングするということになっている。したがって，Z'sTRIPHONYをお持ちでないユーザーは，この機能を利用することができないので悪しからず。

Z'sTRIPHONYは，本誌1989年10月号で一度紹介しているポリゴンのモデリング/レンダリングを行うシステム。操作性は合格点。レビューはサンプル版でのものだったが，製品版ではもっとましな仕様になっていて，32768枚のポリゴンまで処理できるという。モデリングの基礎は確実に学べるので，興味のある方は一度使ってみることをお勧めする。

なおポリゴン関係は本誌ではDōGAのほうがメジャーである。こちらのポリゴンデータも読み込むことができたら2倍おいしかったのだが，残念ながら開発が間に合わなかったようである。そういう動きはあるようなので，次期バージョンに期待することにしよう。

まずは環境設定から

まず，環境設定であるが，最低でもメインメモリ2Mバイトは欲しいところである。CGというのはとにかくメモリを喰う。プリミティブの数は多ければいいというものでもないが，よく作り込んである作品は立派に見え，そうでない作品は手抜きの感を免れないことが多い。結局は作り手のセンスの問題なのだろうが，感性の流れを妨げないためにも，メモリは海よりも広く，が理想である。

メモリを確保するためには，CONFIG.SYSやAUTOEXEC.BATを書き直しておくことになる。たとえばRAMディスクは必要ない。モデリング中はディスクアクセスが頻繁にある関係上，確かにRAMディスクはあったほうがありがたいが，どうしても必要なものではない(僕はハードディスクを使っているが，腹を立てるほどの遅

さではない)。しかしレンダリングのときは，RAMディスクの領域を開放してメインメモリに回したほうが賢明であろう。物体数が制限されるのではたまらないし，マッピングのデータはメモリ上に置く(後述)ので，メインメモリが不足するとえらいことになるからである。というわけでメインメモリの確保は最優先。ほかには，ASKやOPMドライバがいかにもメモリを喰いそう。OPMドライバはメモリ容量だけでなく，レンダリングの速度にも影響しそうなので，涙を飲んで切り離しておくべきであろう。ビジュアルシェルもいらない。サイクロンのシステムはビジュアルシェルから起動するようにはできていない(不可能ではないが)。サイクロンを使いたいならコマンドモードには慣れておく必要がある。

なんだか，ちっともX68000らしくない操作環境なので悲しくなってしまうが，とりあえずそういうことで始めよう。

おっと，「私はメモリを10Mバイト持っているぞ」というリッチな人には，上でいったような注意はまるっきりの無駄です。遠慮なく快適な操作環境で作業してください。

お次はサンプル作成だ

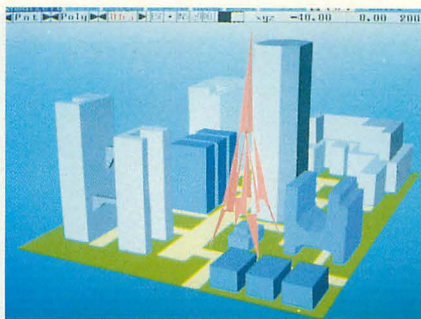
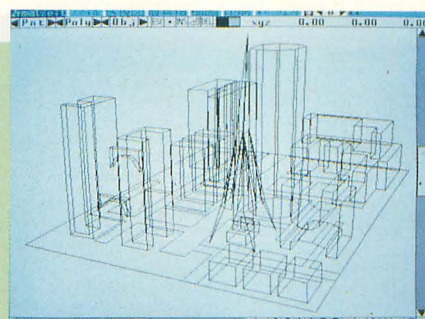
それでは簡単なサンプルを例に取って，サイクロンでポリゴンの作品ができるまでの作業の流れを見てみよう。街の中にタワーが立っているのがそれ。ちょっと色彩感覚が変だが，これはサイクロンのせいではない。僕のセンスの問題です，ハイ。

1) モデリング(Z'sTRIPHONY)

今回はZ'sTRIPHONYを使うことがメインではないのだが，ポリゴンはこちらで作っておく必要がある。

ただ，作業にあたっては，サイクロンへコンバートすることを考えて，少々気を使う必要がある。

サイクロン側ではオブジェクト単位で色をつける。そのため，オブジェクトテーブルに載せるオブジェクトは1色につきひとつにすること。もし，どうしてもひとつの物体が2色以上になるのであれば，各色ごとのパーツに分解し，別々にオブジェクトテーブルに載せること。オブジェクトにはそれぞれ名前をきちんとつけておくこと。

ファイルを保存するときは，普通のフォーマットでセーブしても受け付けない。普通の3D標準ファイル(拡張子.PNX)ではなく，3Dテキストファイル(拡張子.PNA)でセーブしなくてはならない。サンプルのファイル名はTOWN.PNAとしよう。データはBドライブに入れておく。

Z'sTRIPHONYの元データ

Z's TRIPHONYのデータをコンバート

2) ポリゴンデータのコンバート

Z'sTRIPHONYを終了したら，コンバータその1(PN2PL.X)を立ち上げる。

A> PN2PL B:TOWN

起動して少し待つと，ファイル中のオブジェクトをすべて展開して，

object name is 'GROUND'
object name is 'TOWER'
object name is 'BUIL1'
object name is 'BUILS2'
⋮

というメッセージが出て，終わったあとは

B : GROUND. pol
B : TOWER. pol
⋮

というファイルができているはずである。

次はデータ変換の第2段。いま作ったファイルをひとつひとつ，サイクロンが読み込めるフォーマットに変換する。コンバータの名前はPL_SRT.X。

A> PL_SRT B : GROUND
A> PL_SRT B : TOWER
⋮

起動すると，各オブジェクトの頂点数などを知らせるメッセージが出て，

B : GROUND. pul
B : TOWER. pul
⋮

というファイルができる。

3) モデリング(サイクロン)

ここでおもむろにサイクロンのモデラーEDX.Xを起動する。

A> EDX

メニュー画面が出るので，EDITモードに入る。ここで注意しておきたいことは，LOADコマンドでは読み込まないということである。EDITモードの中で1つひとつ読み込む必要がある。具体的には以下の操作をオブジェクト数だけ繰り返すことになる。

プリミティブを作る(MAKE PRIM)。プリミティブのタイプはポリゴンにする。すると「プリミティブの大きさ(SIZE)」という項目の代わりに「ポリゴンのファイル名(POLY FILE)」を指定するように指示される。ディレクトリのウィンドウが開くので，カーソルキーで選ぶとデータを読み込む。あとは普通のプリミティブを定義するのとまったく同じように色つけ，移動，マクロ化を行うことができる。

少ししつこく書いてしまったが，操作が繁雑だ，と文句をいいたいわけである。

それはさておき，今回のバージョンからは，光源の編集をEDX.X上でできるようになった。これまでは別の光源エディタ(LED.X)を起動する必要があったので，光の当たり具合などはレンダリングするまでわからなかった。レイトレースには時間がかかるので，不安になりながら出来上がりを待たなくてはならなかったが，光源エディタをモデラーに組み込んだおかげで不安がなくなった。ワイヤーフレーム表示のときに，光の方向を示す矢印が画面に出てくれるので，光の編集がとてもやりやすくなった。

4) レンダリング

レンダリングの手順はこれまでと同じで

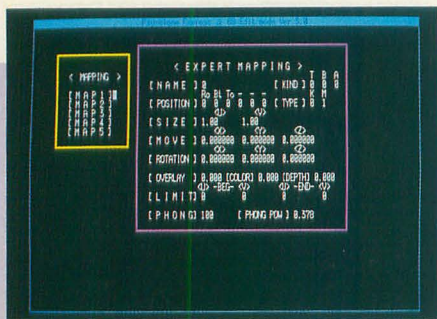
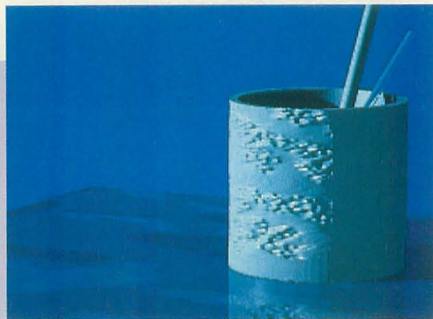
A> VOX B : TOWN

とすれば，ボクセル分割などの前処理をして，ディスクの領域確保のあと，レイトレースを始める。このディスクの領域確保というのは気がきいていて，レイトレース途中でディスクフルになるのを防いでいる。

これは前バージョン(Express)からなのだが，レンダリングの中断・再開ができる。ということは，X68000に昼はゲームの相手をさせ，夜は毎晩レンダリングをさせる，ということもできる。大河ゲームをひとつやりあげる頃にはCGのほうも完成することであろう。大作は夜作られる。

計算が終わると，レンダリングにかかった時間をファイルで出してくれるのも小さな親切。あとで，

A> TYPE B : TOWN. RPT

とすれば，いつでも計算時間がわかる。できた画像は，PDISP.Xで表示できる。

マッピングはExpert Mappingで行う

レンダリングした結果である

気になるレンダリング時間は，演算ドライバFLOAT2+.X，182ポリゴン，光源数1，マッピングなし，256×256ピクセル，アンチエリアシング4×4オーバーサンプリングで12時間30分であった。Z'sTRIPHONYでは数分で済んだことを考えると，さすがに遅い。しかし色指定などがまずかったのだろう，レイトレーシングだからといって格段にリアルになったようにも見えないのが残念である。ポリゴンデータをサイクロンにコンバートするときは，よほどのリアリティが出せるようでないと割が合わないということであろうか。

新しい機能は？

今回のバージョンアップでもうひとつ嬉しいのはマッピング関係のチューンアップである。旧バージョンでは，マッピングの座標系が自由に選べなかった。たとえば球体なら極座標系，直方体なら直交座標系，といった具合に固定されていたのである。通常ならそれでも支障はないのだろうが，細かい指定を行いたいという要望が強かったのだろう，新バージョンでは，Expert Mappingなるモードを新たに用意した。テクスチャマッピング，バンプマッピング，アトリビュートマッピング，それから多重マッピングは前バージョンから可能だったので，事実上あらゆる方法でマッピングができるようになったといえよう。

そうそう，マッピングはプリミティブ数を節約してなおかつリアリティを増すためのおいしい手段でもある。プリミティブ1個とマップ1枚だけで，プリミティブ数十個にも勝る表現力を得ることも可能である。それと，マップは100Kバイト単位でメモリを喰っていくので，もしメインメモリに収まらなければ，ディスク上に置いてレンダリング中に頻繁に読みにいくことになる。これは著しい速度の低下を招くことになるので，多少の無理はしても自由に使えるメモリは残しておくべきだろう。マップをメモリに載せてしまいさえすれば，RAMディスクほどの高速外部記憶はサイクロンには必要ないようである（もちろんあるに越したことはない)。たとえハードディスクがなくても，フロッピーディスクでも計算結果（つまりレイトレース画像）をファイルに出すだけだからレンダリング速度の足を引っ張るほど遅くはならない。しかし，マッピングデータをフロッピーに置いたら，これは悲惨である（初期バージョンではマッピングのデータはファイルでアクセスするということになっていたので，逆にRAMディスクにマップデータを載せておくことが必要であった)。

それからマッピングの品質も上がっているとのことである。マッピングのデータは画像から切り取ってくるので，マップといえどもやはりピクセル単位で色がついている。だから単純にマッピングすると，もとのピクセルの形が物体上に浮かび上がってしまう。このために，注意しておかないと，妙なところでリアリティを損ねる結果となる。典型的な失敗例は，1988年10月号でC-TRACEをレビューしたときのマッピングのサンプルの一部に見られる。物体の表面に網目模様が浮き出ている（お断りしておくと，これは使ったマップが狭すぎたせいで，マップデータをもう少し大きく取れば解消できる問題である。このことをもってC-TRACEの能力を云々することはできない)。で，サイクロンではマップデータに巧みなフィルタ処理を施して，この問題を解消したようである。もっとも今回のサンプルではマップを大きく取ったので，フィルタリングの効果を示すことができなかった。

例によって簡単なサンプルを作ってみよう。テーブルの上に乗っている筆立てのようなものがそれである。テクスチャマッピングとバンプマッピングをそれぞれ1枚ずつ用いている。それでは始めよう。

1） マップデータ作成（Z'sSTAFF）

テーブルの模様はテクスチャマッピングなので，普通に色を使って描けばよい。筆立ての模様はバンプマッピング。バンプマップデータは，画像の赤成分だけを取るようになっているので，初めから赤の濃淡で描いても，白黒でもいい。もしくはスキャナなどで取り込んだ絵をテクスチャマップ・バンプマップ両方に使ってもいい。

2） マップデータの登録

テクスチャカッター（TXCUT.X）を使って，TEXTURE.rgb，BUMP.rgbなどの画像データを作る。マップの大きさは，64×64，128×128，256×256ドットのうちから選べる。このサンプルでは256×256ドットのマップを2枚作った。

3） モデリング

マッピングはさっそくExpert Mappingを使ってみた。初めにプリミティブ上でマップの位置を合わせてしまえば，あとはプリミティブやマクロを動かすとマップのほうでついてくるようになるので安心だ。

4） レンダリング

レイトレース前に，マップデータをメモリに格納する。メインメモリが足りなければディスクアクセスに切り換わるので注意しよう（ちなみにPC-98版だとオンメモリになるマップの大きさには制限があるそうである。セグメントのせいかな？)。

*

レンダリング時間は，演算ドライバFLOAT2+.X，プリミティブ数5，光源数1，マッピング2枚，256×256ピクセル，アンチエリアシング4×4オーバーサンプリングで6時間であった。

マッピングは手軽に使えるわりには，妙に，といったら変だが，表現力の上で得をする。マッピングなしではなんということもなかったシーンが，ちょっとマッピングをかけただけで，いとも簡単に写真のように（大袈裟かな）リアルになってしまうのである。繰り返しておくが，このサンプルではプリミティブの数はたったの5個である。マッピングは効果的に使おう。

ほかにもずるい手段として，物体や背景の色を暗めに設定するというのがある。変に明るいと，画像が白けた雰囲気になってしまうのである。いい表現ができる自信がないなら，とりあえずはこうやってごまかすのも手であろう。実際，結構効果がある。

ここで使用感など

今回はZ'sTRIPHONYとサイクロンを同時進行で使ったので，両者の差が際立ったというのが正直な感想である。

・モデラーを比較する

使ってみた印象をひとことでいえば，Z'sTRIPHONYはPC-98版を引きずり，サイ

クロンはPC-98的なものを引きずっているというところか。

Z'sTRIPHONYは操作体系が，PC-98版とまったく同じである。Z'sSTAFF PRO-68KがPC-98版のしがらみをいっさい断ち切ったのと対照的である。そしてサイクロンのモデラーは，初期バージョンからそうだが，お世辞にもX68000的とは呼べない。なによりPC-98的なのが，階層型のポップアップメニューである。リターンキーでプリミティブを指定し，設定項目を指定するたびにウィンドウがひとつ開く。詳細な指定をするたびにウィンドウの階層が深くなっていく。指定が終わったら，今度はルートまで抜けていかなくてはならない。ESCキーでは1段だけ，UNDOキーではルートまで一気に抜けるという技が使えるおかげで操作性はそれほど悪くないが，なにかほかにいい方法があるのではないかな，などと思ってしまう。

X68000はマウスを標準装備し，PC-98版は基本的にマウスがオプションである。それが両者のソフトウェア環境にどのような影響を及ぼしたかは推して知るべしだが，サイクロンのモデラーともなればマウスの活躍する余地はいくらでもあるはずである。正直いって（失礼ながら）Z'sTRIPHONYのモデラーのほうが「数値」をあまり意識する必要がないので使っていて楽しい。プリミティブのサイズ，マクロの移動，もっと視覚的にできると思う。X68000的なX68000版の登場に熱い期待を寄せたい。

Z'sTRIPHONYからサイクロンにポリゴンデータを持ち越す際にも若干の不満が残る。まず，ポリゴンデータのうち，形状に関係した部分しか持ってこないようだ。つまり頂点のデータだけで，色のデータは持ち越さない。色やアトリビュートはサイクロンのモデラーで改めて指定しなおすことになる。Z'sTRIPHONYとサイクロンでは色のつけ方がまったく違うし，マッピングもあることを思えば，やむを得ないのかもしれない。

が，問題はまだある。それは操作が繁雑だという点だ。これは操作性にひどく影響する部分だといえよう。というのも，Z'sTRIPHONYのファイルを読み込んで変換すると，ポリゴンのファイルがたくさんできる。建物1個で1ファイル，タワー1個で1ファイル，要するに物体ごとに別のファイルを作ってしまうのである。そしてサイクロンのモデラーに読み込む操作もファイルの数だけ繰り返しである。ひたすら機械的な作業である。単調な作業は操作ミスの原因にもなる。

さらに，色は物体ごとにしかつけられない。たとえば屋根だけが赤，ほかは白という家を作りたいとすると，Z'sTRIPHONYの段階から，家の本体と屋根を別の物体としてモデリングしておかないと，望みどおりの色をつけられない。なんとなく二度手間的な作業が多いのである。

ではどうすればよいのだろう。基本的には，一括して読み込む。むしろ，しゃにむに読み込んでもらうくらいがいい。形状データだけでなく，色データからなにから貪欲にデータを飲み込む。そしてそのままで一応レンダリングまではできるようにしてほしい。Z'sTRIPHONYからのデータだけではアトリビュート指定が足りない（たとえばハイライトはZ'sTRIPHONYにはない概念）というなら，勝手にデフォルトの値をつけてしまってもよいではないか。どうしても細かい表現や指定をしたいときに初めてサイクロンのモデラーを起動するくらいの気持ちで作ってもよさそうである。まあそれは極端としても，現段階では手作業でやっている操作の大部分は，プログラムで自動処理できそうな気がする。

・レンダラを比較する

Z'sTRIPHONYのレビューをしたとき，なんて遅いんだ，と思っていたが，今回サイクロンと並行して使ってみて，いかにポリゴンレンダラが速いかを思い知らされることになった。皮肉なものである。

それでもあえてポリゴンデータをレイトレーシング向けにコンバートするメリットは何か？　それは，影・マッピング・反射・屈折といった，レイトレーシングが得意とする表現であろう。これらのおかげで平板なポリゴンにリアルな表情がつく。逆にいえば，レンダリングの速度や手間の繁雑さといったデメリットを上回るだけのメリットが見出せないのならば，Z'sTRIPHONYで通したほうがいいのかもしれない。

ところで，PC-98版だが，トランスピュータボードを装備した「サイクロンExpressタキオン（いかにも速そうな名前ですな）」というのも発売された模様である。はたしてX68000へのサポートは？　ユーザーの要望が多ければ実現するかも。

完璧なハードコピーが欲しいあなたへ。GP-IBボードを介してPIXEL DiOにデータを送れば，フルカラーの画像が見られる。確かにX68000は65536色しか画面には出せないが，ちゃんと1670万色になるように計算だけはしているし，ファイルにもそのフォーマットで出力している（同じドット数でもファイルサイズは1.5倍になる）。だから，しかるべきところに持っていけば，フルカラーのハードコピーも取れる。その気になれば，512×512以上の巨大な画像のハードコピーを取ることも可能だろう。

まとめ

ずいぶんと苦言も呈してしまったが，サイクロンはデビュー時からはるかに進歩し，使いやすくなった。文句は，もっとよくなってほしいという希望の裏返しである。

それからユーザーの方には，まずはじっくりと使い込んでほしい。3次元CGに限らず，一般論で，まともな作品を作ろうと思ったら，まず道具の使い方を身につけることである。3次元CGの場合は，道具が複雑怪奇で取っ付きにくいという特性があるが，そんなことでくじけてはならない。モデラーを使うだけで精一杯というのではまだまだで，手足のように使いこなせるようになって初めて，創造性などといったレベルの話ができるのではなかろうか。「これこれこんな機能を使ってみました」みたいな主張が見え見えの作品は，学会で新手法を発表するときや，新作ソフトウェアのレビュー記事にはいいかもしれないが（う～むさりげない自己弁護），そこから抜け出せない限り明日はないであろう。

今年の4月号から本誌の表紙を飾っているCGは，もうご存じの方も多かろうが，レイトレーシングである。それも決して超高級機ではない，ごく普通のパーソナルコンピュータで計算することができる（している）。もはやパーソナルコンピュータは，十分に優れた道具たりうる。

いま，道具はあなたの手にある。生かすも殺すもあなた次第である。

Z'sTRIPHONY DIGITAL CRAFTとは

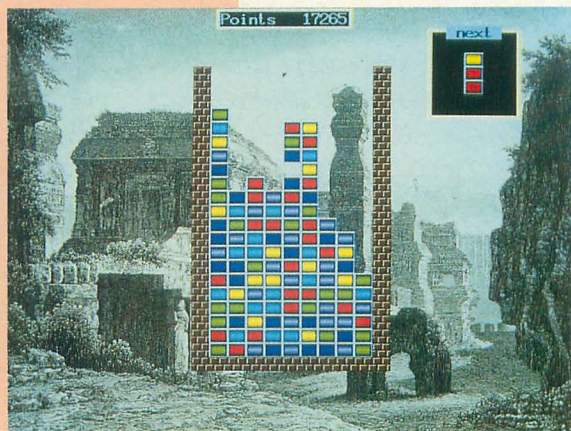
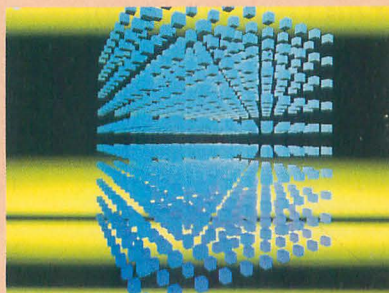
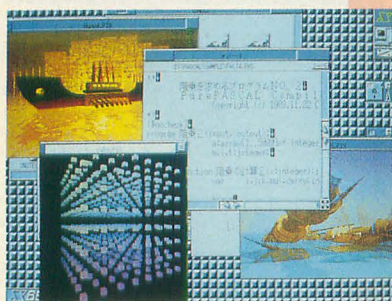
Z'sSTAFFのツァイトとトリフォニー（だいぶ昔にX1版があった）のアーマットが作った3次元CGソフトウェア。ポリゴン（多角形）の集まりでものを表現する。システムはモデラー（設計をする部分）とレンダラ（描画をする部分）からなる。

操作はマウスで行い，慣れれば感覚的に扱えるようになる。回転体や面掃引などの機能を使えば面数の多い立体も簡単に作れる。ひとまとまりのポリゴンをオブジェクトとして管理できるので，オブジェクト単位の移動・複写も簡単。複雑な形も比較的簡単に作れる。扱えるポリゴンの数も多いので不自由することはない。

フラットシェーディングだけでなくスムーズシェーディングも可能なので，ポリゴンを疑似的に曲面に見せることもできる。

全体的に扱いやすく仕上がっているCGツールだ。詳細は1989年10月号にレビューがある。

特集

創刊8周年記念PRO-68K

X68000ユーザーも10万人を目前とし，ようやく市民権を得たパソコンとして認められるようになりました。市販ソフトの状況も概ね好調。この機にOh!X初の試みとして2HDディスクの付録を企画しました。
基本的に雑誌に掲載されたプログラムは読者の手で打ち込まれるべきだ，とOh!Xは考えます。すべてが打ち込まれるべきだ，とはいいません。多くはアルゴリズムの参考程度にとどまるでしょう。
もし誰もリストを打ち込まなくなったら，もし，パソコン雑誌からプログラムリストが消えたら，多くの人はパソコンの本当の面白さを知らないままに過ごしてしまうような気もします。意志を持ってアクティブに動くユーザーがいないパソコン界などなんとつまらないことでしょう。
にもかかわらず，Oh!Xが提供しようとするのが今回のディスクです。
今回のディスクに収録されたプログラムはX68000ユーザーほかが築き上げてきたもののほんの一部にすぎません。そして，これらのツールはさらなる次元へ私たちを誘ってくれると確信します。

なにが入っているのか

では，Oh!Xが220円の値上げと派手な予告をしてまで断行したオマケディスクとはいったいなんなのでしょうか？

このディスクには，

●フルセット仕様PASCALコンパイラ
PurePASCAL
●ワークステーションの高性能C
GNU Cコンパイラ
●超高性能ソースジェネレータ
DIS.X
●対話型3D CGAシステム
ANGEL
●Oh!X標準ミュージックドライバ
OPMD.X & オリジナルPCMファイル
●25トラック対応ミュージックドライバ
MUSICDRV.X
●ウイルス検出プログラム
DOCTOR.R
●高速数値演算ドライバ
FLOAT2+/3+.X
●ブロックゲーム
Yet Another Column
●全機種共通システム
X68000対応S-OS“SWORD”
PC-286対応S-OS“SWORD”
●コマンドシェルシミュレータ
X1turbo用INTEGRAL X1

ほか，多数のユーティリティ群が収録されています。どうせ滅多にできないんだからと，雑多なツールを放り込みましたが，基本的に，打ち込み作業の限界を超えたツール群を配布することが第一の目的です。

では使い方

ここに収録されたプログラムのほとんどがLH.X（50ページ参照）によって圧縮されています。これらを使用するには以下の手順でファイルを解凍（展開）することが必要です。

まず，手持ちのシステムディスクを立ち上げます。そして，戦車のアイコンをダブルクリックするなどして，コマンドモードに入ってください。

次に表1を見て解凍したいプログラムがどのファイルに圧縮されているかを確認します。

解凍にはLH.Xを使用します。LH.Xは特に指定しない限り，カレントディレクトリにファイルを展開しますので，たとえば，Aドライブにフォーマットしたばかりのディスクを入れ，オマケディスクをBドライブに入れてください。ここでPASCALを解凍する場合なら，

A>B：LH －E B：PASCAL

のような操作をします。“.LZH”という拡張子のついたファイル1つひとつに対して，同様の操作を行うわけです。

なにも考えずにまとめて展開したいという場合には，付属のDISK.BATを使用してください。手順は以下のとおり。

1) 2HD生ディスクを3枚フォーマットする
2) システムディスクから起動し，コマンドシェル（COMMAND.X）を立ち上げる
3) オマケディスクをAドライブに入れ，Bドライブにフォーマットしたばかりのディスクを入れる
4) 次のようにバッチファイルを起動する

A>DISK B:

5) 指示があればディスクを入れ替える

6) あとは待つだけで展開が終了する

このときHuman Ver.1ではバッチのためのメモリが足りませんので，コマンドシェル起動時に，

A>COMMAND /B:2

のようにオプション指定し，数値演算ドライバは必ず組み込んでおいてください。

これで，ほとんどのファイルが3枚のディスクに転送されました。が，ちょっとした手違いで，SMPL（サンプリング音を集めたもの）が展開されずに残っていますので，手動で展開してください。手順は，

A>B:

B>RD SND

B>MD SMPL

B>CD SMPL

B>A:LH -E SMPL

以上です。

2HDドライブをお持ちのX1turboユーザーの方は55ページの手順でINTEGRAL X1のプログラムが取り出せます。

注意点

もう一度注意点をまとめてみましょう。

●システムは入っていない

立ち上げには手持ちのシステムディスクが必要です。

Human Ver.1の場合はそのままでは展開用のバッチファイルが実行できないので注意してください。

手順でよくわからないところがあったら各自のマニュアルをお読みください。

●主要なファイルはすべて圧縮されている

解凍には付属のバッチファイルを使ってまるごと展開するか，LH.Xを使ってひとつずつ必要なものを展開していくことになります。

●バグがないとはいえない

本誌自体よりディスクの締め切りが早かったため，すでに判明している誤りだけでも以下のようになります（慣れないことをするといけませんね）。

1) 封筒のMUSICDRV.Xの部分がMIDIDRV.X Ver.2.0となっている。

2) FLOAT2+.Xの動作がおかしくなることがある（たまに計算を間違う）。ANGELは実行しないように。

3) DOCTOR.Rのエラーチェックが甘い。誤操作でウイルスの侵入を許す（要注意）。

4) DISK.BATでSMPL.LZHが解凍されない。

5) X68000以外の機種用のファイル解凍ツールがついていない。

主要な収録プログラムについての解説は特集記事，それ以外のプログラムデータなどについては過去の掲載記事などを参照してください。

*　　　*　　　*

今回のディスク作成企画に対して快くプログラムを提供してくださった方々，イマジネーションを裏切らない素晴らしいマシンを作られたX68000の開発者の方々，わがままなユーザーを辛抱強くサポートしてくださったソフトハウスの方々，そして8年のあいだOh!Xを愛読してくださった読者の方々に感謝いたします。

表1　ディスクの内容

ディレクトリ	ファイル	形式	フリー
INTEGRAL		X1	*
LH		X	*
TOOLS		LZH	
	DOSCALL.MAC		
	IOCSCALL.MAC		
	PIC.FNC		*
	PICLIB.a		*
	lh.doc		*
	sxconv.x		*
	opmsco.x		*
	PIC.R		*
	MAC.X		*
	SPR.FNC		*
	vprint.x		
	remocon.fnc		*
GCC		LZH	
	gbc.bat		*
	gcc_cpp.x		*
	gcc_cc1.x		*
	knj2oct.x		*
	GNULIB.A		*
	GNU.ME		*
	readme		*
	gcc.x		*
SWORD		LZH	
	TRANS.S		*
	PRINT.S		*
	Z80.S		*
	SWORD.S		*
	TRANS.X		*
	SWORD.X		*
	SWORD.EXE		
DIS		LZH	
	dis.x		
	readme.doc		
SYS		LZH	
	FLOAT3+.X		
	float2+.x		
DOCTOR		LZH	
	doctor.r		*
	tfr2.x		*
	zap.x		*
ANGEL		LZH	
	angel.x		
	cubic.ang		
	cube.mdl		
	readme.doc		
	usage.doc		
	fighter.ang		
	f_fin01.mdl		
	f_body01.mdl		
	f_nozl01.mdl		
	f_tank01.mdl		
	f_tank02.mdl		
	f_tank03.mdl		
	f_canop.mdl		
	f_nozl02.mdl		
OPMD		LZH	
	OPMD.x		*
	MT32		*
	m1		*
	k1		*
	d10		*
	KAEND.bas		
	kaend		*
	kaend.bat		*
	kemt		*
	kemt.bat		*
GAMES		LZH	
	siro.gs3		*
	99.BAS		*
	FONT.DAT		*
	MAKE.bas		*
	yet.x		*
MUSICDRV		LZH	
	MUSICDRV.x		*
	MUSIC2.FNC		*
	SOUND.SND		*
	MUSIC.doc		*
	DEMO.BAS		*
	MT32.SND		*
PASCAL		LZH	
	PASCAL.X		
	PASLIB.A		
	PASCAL.MAN		
	PASCAL.MAC		
	PL0.PAS		
	GTEST.PAS		
	SIEVE.PAS		
	FACT2.PAS		
	RECER.PAS		
	STRTEST.PAS		
	PI.PAS		
	GRAPHIC.LIB		
	BENCH.INC		
SMPL		LZH	
	KICK.PCM		*
	BD2.PCM		*
	TOM1_1.PCM		*
	TOM1_2.PCM		*
	TOM1_3.PCM		*
	TOM1_4.PCM		*
	clp.pcm		*
	whip1.pcm		*
	whip0.pcm		*
	BD3.PCM		*
	BD4.PCM		*
	RIDE1.PCM		*
	crsh2v13.pcm		*
	SIDE.PCM		*
	HHC1V8.PCM		*
	HHO1V8.PCM		*
	HHC1V10.PCM		*
	HHO1V10.PCM		*
	CRSH2V8.PCM		*
	CRSH2V10.PCM		*
	HAMMER.PCM		*
	SNAP.PCM		*
	WIND.PCM		*
	SD2V10.PCM		*
	SD2V13.PCM		*
	SD2V15.PCM		*
	SD3V10.PCM		*
	SD3V13.PCM		*
	SD3V15.PCM		*
	BS_SD.PCM		*
	SD5V10.PCM		*
	SD5V13.PCM		*
	SD5V15.PCM		*
	ET1V14.PCM		*
	ET2V14.PCM		*
	ET3V14.PCM		*
	ET4V14.PCM		*
	ET1V10.PCM		*
	ET2V10.PCM		*
	ET3V10.PCM		*
	ET4V10.PCM		*
	ET1V8.PCM		*
	ET2V8.PCM		*
	ET3V8.PCM		*
	ET4V8.PCM		*
	SPSND.PCM		*
	TIMB1.PCM		*
	TIMB2.PCM		*
	TIMB3.PCM		*
	TIMB4.PCM		*
	sd_cymv8.pcm		*
	SD_CLP.PCM		*
	BD1V10.PCM		*
	BD1V12.PCM		*
	BD1V15.PCM		*
	SD_ETM.PCM		*
	BD_HC.PCM		*
	BD_HO.PCM		*
	SD1V10.PCM		*
	SD1V12.PCM		*
	SD1V15.PCM		*
	CONGA1.PCM		*
	CONGA2.PCM		*
	TOM2_1.PCM		*
	TOM2_2.PCM		*
	TOM2_3.PCM		*
	TOM2_4.PCM		*
	TOM2_5.PCM		*
	Oh_3F.PCM		*
	Oh_3F#.PCM		*
	Oh_3G.PCM		*
	Oh_3G#.PCM		*
	Oh_3A.PCM		*
	Oh_3A#.PCM		*
	Oh_3B.PCM		*
	Oh_4C.PCM		*
	Oh_4C#.PCM		*
	Oh_4D.PCM		*
	Oh_4D#.PCM		*
	Oh_4E.PCM		*
	Oh_4F.PCM		*
	Oh_4F#.PCM		*
	Oh_4G.PCM		*
	Oh_4G#.PCM		*
	Oh_4A.PCM		*
	Oh_4A#.PCM		*
	Oh_4B.PCM		*
	Oh_5C.PCM		*
	Oh_5C#.PCM		*
	Oh_5D.PCM		*
	Oh_5D#.PCM		*
	Oh_5E.PCM		*
	BD_SD.PCM		*
	RIDE2.PCM		*
	TOM1_1V8.PCM		*
	TOM1_2V8.PCM		*
	TOM1_3V8.PCM		*
	TOM1_4V8.PCM		*
	tm1_1v13.PCM		*
	tm1_2v13.PCM		*
	tm1_3v13.PCM		*
	tm1_4v13.PCM		*
	sd_cyv13.PCM		*
	bos		*
	BD1.PCM		*
	ASN.PCM		*
	CHH.PCM		*
	OHH.PCM		*
	TAMB.PCM		*
	MARACAS.PCM		*
SWORD_2D		LZH	
	sos.d		

各プログラムの著作権はプログラム開発者に保留されています。
＊印はフリーウェアとして扱ってかまいません。
プログラムはC compiler PRO-68K, GCC, X-BASICなどで開発されたものです。

Tools 便利なツールあれこれ

編集部

ディスクに収録されたプログラムの解説です。手始めに小さなツール群からいってみましょう。ディスクではTOOLS. LZHなどに収録されています。これまでOh!Xに掲載されたものがほとんどですのでお馴染みのものもあるかもしれませんね。興味を持った方はぜひ使ってみてください。

ディスクに含まれる小規模なツール群をまとめて紹介する。これまでに掲載されたことのあるものに関しては掲載号も参照してほしい。

LH.X

定評のあるファイル圧縮ツール。オリジナルは吉崎栄泰氏でちー君氏がHumanに移植したものを収録した（詳しくはドキュメント参照)。アルゴリズム解説はまたの機会に譲るが68000のオブジェクトコードを半分近く，テキストデータなら2，3割の大きさに縮めてしまう驚異の高圧縮書庫管理プログラムだ。今回のプログラム群がディスク1枚に収まったのも，ひとえにこのプログラムのおかげといえる。

基本的な使い方を説明しよう。

AAAAというファイルとBBBBというファイルをまとめて圧縮したいなら，

```
LH-AH CCCC AAAA BBBB
```

のようにする。ディスク上には両者が圧縮された“CCCC.LZH”というファイルができる。

これを解凍するには，

```
LH-E CCCC
```

のようにする。これでカレントディレクトリ上に展開されたファイル群が生成されることになる。

PIC.R

ご存じ，柳沢明氏による65536色用高性能グラフィック圧縮ツール。X68000ではもう標準となった感もある。

セーブ時は，

```
PIC/S　ファイル名
```

ロード時は，

```
PIC　ファイル名
```

のように使用する。

稲妻走るアルゴリズムの解説は1990年2月号参照。まだお持ちでない方はぜひ使ってみてほしい。

PIC.FNC

この関数をBASICのディレクトリに入れ，BASIC.CNFに，

```
FUNC=PIC
```

の1行を追加することにより，

```
pic_load( )
pic_save( )
```

の2つの関数が拡張される。

だいたいおわかりと思うが，これはimg_load( )，img_save( )の代わりにPIC形式で画像のロード/セーブを行うための関数だ。

使用については多少制限があり，元々のpic.rと同様，画面モードが512×512，65536色モードのときの32768色にしか対応しない。具体的な画面モードでいえば，

```
SCREEN 1,3,1,1
SCREEN 0,3,1,1
```

などだ。

ロード時にロード座標が指定できることなどは従来のimg関数と同様だが，セーブの際に矩形の範囲指定が自由にできるように拡張されている。画面モード制限と256×256モードのときのセーブサイズ以外にはあまり気をつけることはないだろう。

続いて，これを使って作ったプログラムをコンパイルするときに使うのがPICLIB.Aだ。コンパイル時に，

```
CC TEST.BAS PICLIB.A
```

のように指定する。面倒ならば，PICLIB.AをBASLIB.Aにアーカイブしてしまうのがよいだろう。もちろん，BASICから使うだけでなく，そのままC言語からライブラリを呼び出しても構わない。

OPMSCO.X

OPMファイルをMUSIC PRO-68Kの楽譜ファイルに変換するためのプログラム。1990年3月号に掲載されたプログラムより多少バージョンが上がっている。

使用法は，

```
OPMSCO [-スイッチ] OPMファイル名
```

すべてのOPMファイルを変換することはできないが，込み入ったOPMファイルもループを展開したりと，かなりがんばって変換している。SCOファイルの詳細は1990年3月号を参照のこと。

VPRINT.X

ビデオプリンタ活用プログラムの決定版。コンピュータコントロール端子での直接接続，加えて文字領域まで画像用に使うことで用紙一杯の画像出力が可能。

なお，使用にはカラービデオプリンタCZ-6PV1，および*.GL3形式の画像ファイルが必要。

MAC.X

Oh!X標準MACINTO-C形式のマシン語入力ツール。1990年3月号で掲載したバージョンをコンパイルしたもの。ファイルエディタとして使えなくもないが，64Kバイト以上は扱えないので注意。

SPR.FNC

ショートプロぱーてぃで掲載された，スプライト衝突判定用外部関数。

```
SP_CHK(A, B, C)
```

のように使うと，スプライト番号Aのスプライトがスプライト番号B～Cまでのうちどのスプライトと重なっているかを返す。1990年4月号掲載のもの。

REMOCON.FNC

1989年6月号に掲載された桒野雅彦氏の学習リモコン制御用外部関数。X68000に赤外線センサと赤外線ダイオードをつけて多機能マルチリモコン化し，各種家電機器をコントロールするためのもの。簡単な工作でホームコントロールができる。

リアルタイムパズルゲーム

Yet Another Column

Izumi Daisuke
泉　大介

編集室でテトリス以来の盛り上がりを見せるブロックゲームです。より難解なパズル性とまとめてブロックを消すときの快感がまさに麻薬的。高得点には一瞬の判断力が要求されるぞ。作者は泉大介氏，アルゴリズムについてはX-BASIC入門を参照してください。

それはソフトバンクの某編集部から始まった。古代フェニキアに端を発するといい伝えられるひとつのゲームが，複雑な経路をたどった末に流れ着いたのだ。最初の犠牲者はその編集者たちであった。原稿の仕上がりを待つあいだに手を出した者たちが次々と毒牙にかかったのである。次号の特集の企画，原稿の発注はもとより，睡眠すら忘れて彼らはそこに展開される世界に没頭し，持てる力のすべてを吸い取られ，さながらゾンビのごとく延々とテンキーを叩き続けたのである。

洗脳された彼らは，新たなる犠牲者を求めてクチコミの布教活動を開始した。甘言に誘われて次に手を出したのは，締め切りが過ぎるまで原稿を書き始めないという慢性の病気から，月末には缶詰にされて原稿を書かされる筆者たちであった。彼らはすでに遅れている原稿そっちのけで陶酔し，原稿を書くべき指はテンキーに吸いつけられたまま，1ページ書き進むのに十数時間かかるというありさまであった。それでも本が出たのは奇跡に近い偉業であったといまさらながら思えるほどである。

編集者に誘惑され地獄の缶詰生活を送った筆者たちによって，布教活動はさらに広がった。そのなかにOh!Xの筆者がひとり混じっていたのが悲劇の始まりだったといえるだろう。

当然の如く彼はOh!Xの編集者を巻き込み，いつのまにやらX68000にそのゲームを移植するという計画ができあがっていたのである。TETRISに飽きてきていたという理由もあろう。上から降ってくるタイルを並べて消していくというルールが，似通っていて親しみやすかったという理由もあろう。しかし，これほどまでに人を惹きつけるのは，ほかならぬColumnsに施された古代の魔法の力以外のなにものでもない。連綿と受け継がれて今日まで生き長らえたその魔力が編集部を越え，筆者を越えて伝搬しているのである。そして今回のディスクにもその魔力が封じ込まれている。

ルール

Columnsのルールはきわめて単純で，上から落ちてくる3連のタイルを使って，積もったタイルを消していくだけである。3連のタイルにはそれぞれ色がついている。タイルを消すには，同じ色のタイルを縦，横，斜めいずれかに3つ以上並べればいい。

もちろん複数の方向に重複して並んでいてもかまわない。タイルは全部で6色ある。「4」「6」のキーは落ちてくるタイルを左右に移動するのに，「5」のキーはタイルの順番を入れ替えるのに使う。「5」を押すたびに縦に並んだ3連のタイルは，一番下にあるタイルが一番上になるように回転する。タイルを落とすのは「2」またはスペースである。

YETの起源

Columnsにはすでにいくつかのバリエーションが存在する。ゲームセンターに置いてあるものを見かけた方もいらっしゃるだろう。ネットにアップされたMacintoshからの移植版をプレイされた方もあろう。それぞれに点数の計算方法などに違いがあり，それゆえ，付録ディスクに収録されている本プログラムはYet Another Columnと命名した。基本的にPC版（無論IBM）のPDSである“Beyond Columns”を参考にしつつ作成されている。

そもそも，Columnsというゲームはヒューレット・パッカードのJay Geertsen氏がX-window用に作成したものが最初らしい。その後，Nathan Meyers氏によりIBM PCに移植され，P.Taylor氏がBeyond Columnsに仕上げたのだ。詳しいことはわからないがゲームセンターにあるセガ版では古代フェニキア産のゲームが元だというからほぼ完璧なPDSといえるだろう。

主な相違点は，消したときの状況に応じてボーナスが支給される仕組みになっていることである。高いところから落としてタイルを消すとボーナスとなる。このとき支給されたボーナスは連鎖反応（古代魔力のひとつの具現。これに取り憑かれた者も多い。あるいはタイルが連鎖的に消えていく現象）のあいだも持続する。高得点を狙うなら，タイルは極力高いところから落とすべきである。このあたりはMacintosh版TETRISの影響といえる。

背景など

背景の絵はsiro.gs3に収めてある。これは512×512ドット×16色のファイルで，表示したくない場合はリネームしていただきたい。黒い背景でプレイできるようになる。背景はなんの役にも立っていない。しかし，TETRISの無念を忘れ得ぬ者の陰謀によって強行に導入された。面ごとに絵を変える予定だったが，ディスク容量の関係で断念せざるをえなかった。

また，自分で用意した絵を表示することも可能である。同様の絵を用意し，siro.gs3というファイル名でセーブすればいい。絵はパレットコード0～15の順に明るくなっていく単色16階調で表示される。

上位10位に入った場合は，自分の名前を登録しておくことができる。半角6文字以内で名前を入力していただきたい。オープニング画面で「S」キーを押すと現在のスコアがセーブされる。ファイル名はyetscoで，yet.x，siro.gs3とともに同じディレクトリに収めておく必要がある。プロテクトシールが貼ってあってもエラーメッセージを表示しないので，スコアを残しておきたい方は注意されたい。ちなみに編集室でのハイスコアは40860点（5月1日現在）である。

＊　＊　＊

このゲームをいまはほかの編集部に移ってしまわれた(よ)嬢に捧げる。Columnsをこよなく愛する彼女の熱意と歓声と微笑なくしては，このゲームの完成はありえなかったであろう。

音楽演奏関係

OPMD & MUSICDRV

ディスクには音楽演奏関係のドライバを2種類，両者で使えるサンプリングファイルをどーんと収録してあります。ドライバはどちらもMIDI+AD PCM+FM音源に対応。X68000の音楽環境を考えるうえでの要となるものです。ぜひとも，活用してみてください。

編集部

ディスクにはOh!XオリジナルのOPMD.Xとサン・ミュージカル・サービスの提供によるMUSICDRV.Xの2つのドライバ，両者で共通に使用できるAD PCMリズムライブラリが収録されています。

2つのドライバの機能は似ていますが，それぞれ違った性格を持っています。

OPMDは通常のMIDIの機能のほかにドライバレベルで楽器間の差異を吸収するという実験的な試みを盛り込んでいます。

このドライバはOPMの未使用レジスタを使って，外見上X68000のOPMを“OPMライクな拡張音源モジュール”に変えてしまいます。OPMDRVを使ったソフトはすべて同様の拡張が行われます。たとえば，MUSIC PRO-68Kも最新のSX-WINDOW上のプレイヤー.XもOPMDを組み込むだけでAD PCM+MIDI対応になるのです。

それに対して，MUSICDRV.Xは本格的なMIDIドライバを内蔵音源（FM音源+AD PCM）対応に拡張したものです。MUSIC PRO-68K［MIDI］に付属していたMIDIDRV.Xの機能拡張版といえます。

リズムはもちろん，内蔵音源もMIDIライクに制御できるほか，基本的なMIDI信号の入出力機能をも備えています。提供元のサン・ミュージカル・サービスによれば「Musicstudioくらいのものならこれで作れる」そうです。

OPMDが内蔵音源を拡張しているので8トラックしかサポートしていないのに対し，MUSICDR Vは25トラックのシーケンスが可能になっています。ディスクには，これに対応してBASIC上から25トラックのMMLを使うことのできるMUSIC2.FNCも収録されています。

OPMDに満足できなかった人はMUSICDRVを使用すればよいでしょうし，これまでのOPMデータとの互換性を重視するならOPMDを使えばよいでしょう。

コンピュータミュージックの発展とMIDIの普及が我々の共通の願いです。これらの基本プログラムとデータはフリーウェアとします。自作プログラムはもちろん，市販ソフトを問わず自由に使用してかまいません（ひと言ライセンス表記を入れてください）。

OPMD.X

従来のFM音源演奏を拡張してAD PCMとMIDIへの出力を可能にしたOPMDドライバの最新版です。

3月号で発表されたバージョンとは少し異なります。基本的な仕様はこれまでと変わりませんが，楽器の割り当て部分が拡張され，バグフィックスしてあります。すでにOPMD.Xをお使いの方も新しいバージョンを使用するようにしてください。

サンプルとしてナイトアームズのエンディングテーマが収録されていますので参考にしてください。これは内蔵音源，KORG M1，Roland MT-32対応となっています。専用のコンフィギュレーションファイルを使用してください。

PCMデータとコンフィギュレーションファイルについて

OPMA.X/OPMD.XまたはMUSICDRV.Xなどで使用可能なリズム関係のサンプリングデータを用意しました。使ってください。いろいろ気を使ってサンプリングしたのですが，X68000内蔵のA/Dコンバータが安価なものを使っているせいか，原音の70%程度のクオリティに下がってしまうのは残念なところです。

あと，音量の大きい音は鳴り終わったときに「プチ」というノイズが入ってしまいます。最近，この「プチ」ノイズの入らないPCMドライバ（PDS）が出回っているそうなのでそれらをなんとかして手に入れるといいでしょう（例：NNPCMDRV.SYS 作者GORRY氏）。

「BOS」というファイルは今回収録したサンプリングデータをOPMAとコンパチ，つまり「ボスコニアン」のサンプリングデータの並びと同じにするコンフィギュレーションファイルです。

A>OPMD /A BOS

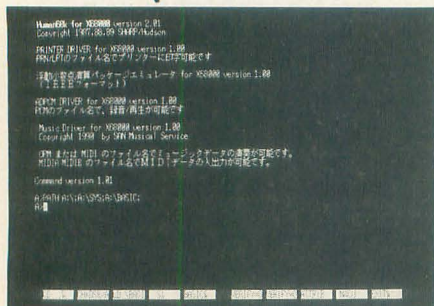
でOPMDを組み込めば，OPMAドライバ発表以後のミュージックプログラムが違和感なく演奏できるはずです。

今後，今回収録したサンプリングデータを使用してミュージックプログラムを制作，投稿する際には必ず一緒にコンフィギュレーションファイルを添えて送ってください。

「M1」や「MT-32」というファイルは見かけ上，KORGのシンセM1，RolandのモジュールMT-32のリズムキットをボスコニアンのサンプリングデータの並びと同じにするファイルです。

M1の場合はプリセットの音色の関係上オーケストラヒットは鳴りません（RolandのサンプラS-330をMIDIチャンネル16に設定しておき，M1のMIDI THRUと接続した場合を想定したファイルになっています。要するにS-330があればオケヒットは鳴るぞということ。あ，ちなみにオーケストラヒットの音色はS-330にもプリセットされてません，別売りです）。サンプラなんかないという人は手近な音（ないと思うが）またはなにか代わりの音で代用するようにしてください（1990年3月号参照）。Tシリーズでサンプリング音を拡張できる（いいなあ）という人はそのように対処してください。

A>OPMD /M M1（またはMT32）

でOPMDを組み込めばOPMAドライバ発表

以後のミュージックプログラムのリズムセクションをMIDI楽器で鳴らすことができます。

今回はMT-32とM1のみですが，順次この手のコンフィギュレーションファイルは発表していきたいと思います。しかし，機材の関係から十分な対応ができない場合がありますので，広く投稿も募集します。お手持ちのMIDI楽器に対応したコンフィギュレーションファイルを作成された方はぜひご一報ください。

MUSICDRV.X

サン・ミュージカル・サービス提供のミュージックドライバです。訂正が間に合わなかったためディスク用の封筒にはMIDIDRV.XのVer.2.0となっていますが，ディスク収録時点ではさらに機能アップしてMUSICDRV.X Ver.1.0となりました。

MUSIC PRO-68K[MIDI]に付属のMIDIDRV.Xは16チャンネルのシーケンスが可能なMIDIドライバでしたが，本体内蔵音源にはまったく出力できませんでした。今回のMUSICDRVではMIDIに16，FM音源に8，AD PCMに1チャンネルを指定して合計25チャンネルの演奏が可能です。単なるMIDIドライバを超えたMUSICDRVのゆえんでしょう。

MIDIはもとよりこうした内蔵音源のサポートに加え，MIDI信号通信のための基本的な入出力機能をも備え，BASICレベルならMIDI信号に時間情報のついたデータの録音までできます(分解能48)。拡張されたシステム予約ファイル名"MIDI"，"MIDIA"，"MIDIB"のファイルに出力することでOSレベルでMIDI信号の送受信が可能です。

MML部分の基本的な使い方はOPMDRVとほぼ同様。拡張機能として，コントロールチェンジの出力やモジュレーション，ベンド，ダンパー用コマンドのサポート，パンポット拡張，1トラック内での和音指定などがサポートされています。

BASICからこれらの機能を扱うMUSIC2.FNCも大幅に機能アップされています。また，MUSIC2.FNCにはなぜか，画面のプライオリティを変更するscr_prw( )関数がついています(BASICでは固定されていたテキスト，グラフィック，スプライトなどの上下関係を変更する)。

そのほか，関数の詳細，アセンブラからの使用法など，詳しくはディスク内のドキュメントをご覧ください。

FM音源ドライバとしてのMUSICDRV

本筋からまったくはずれてしまいますが，もし，このMUSICDRV.XをFM音源+PCM音源ドライバとしてとらえるとどうなるでしょうか？

OPMDRV.XのMMLは制御構造は強力ですが，表現力という面では特に細工をしない素直なMMLです。しかし，現状のFM音源に要求される音はそんなに素直ではありません。結果として現在のOh!X LIVEのようにOPMを直接ドライブするYコマンドが山のように現れ，はなはだしい場合，作った本人にもわからないリストになります。

MUSICDRVではこの伝家の宝刀Yコマンドが禁じられています。では十分な表現力は期待できないのでしょうか？　いえ，そうでもありません。もちろんOPMを知り尽くした者だけに許されるような超テクニカルプログラムはできません。しかし，これまで対応するコマンドがないため，Yコマンドで直接制御していた部分がMMLレベルでもサポートされているのです。

モジュレーションやピッチベンド，さらにはダンパーON/OFFの指定までMIDI楽器と同様に指定可能です。これならYコマンドに頼らなくても，かなりの表現が可能ですからミュージックプログラムもずいぶんすっきりしたものにできるでしょう。

加えて，AD PCMのサンプリング音がノートナンバーに割り当てられました。つまりMIDI楽器のようなリズムキットを構成できるわけです。たとえばバスドラムをCに，スネアドラムをDに，オープンハイハットをEの音に……と割り当て"CDCDCDE"のように演奏するわけですね。一度に使用できるサンプリング音はCからBまでの最大12音に制限されていますが，これはプログラム起動時(ドライバ組み込み時でなく)に指定するものですから1曲あたりなら必要十分といえるでしょう。

内蔵音源ではなくてMIDI楽器を使うような感覚で操作するものなのです。

OPMDのようなテーブルは？

このプログラムはOPMDRV.Xに相当するものですから，さらにOPMDのようなテーブルを持ったドライバを加えることで多機種に対応できるはずです。多くのプログラムはMT-32用に作られると思いますが，対応するコンフィギュレーションファイルを作成することによって共通データを扱うことができるかもしれません。これはこれからの課題です。

*　　　*　　　*

最初にいったとおり，OPMD，MUSICDRVのいずれも特徴を持ったミュージックシステムです。各自の目的に応じて使い分けてみてください。

OPMDでの投稿には使用したコンフィギュレーションファイルを必ず添えるようにしてください。もちろん，MUSICDRVでのOh!X LIVE投稿もお待ちしています。

[OPMDサンプル曲]
ナイトアームズ・エンディングテーマ

©Arsys SOFTWARE

オマケのディスクからOPMDとSMPLを解凍します。たとえば，容量の十分にある(500Kバイト以上)BASICシステムディスクを用意してAドライブに入れ，オマケディスクをBドライブに入れた場合なら，

A>B:LH-E　B:OPMD
A>B:LH-E　B:SMPL

のようにコマンドモードから打ち込んでください。手違いにより，ディスクに付属しているDISK.BATではSMPLが正常に解凍されませんので注意してください。

それではサンプルプログラムを聞いてみましょう。今回も内蔵音源のみ，M1，MT-32にそれぞれ対応しています。

内蔵音源のみで鳴らす場合には，

A>CD SMPL

としたあと，今回収録されているPCMデータを，

A>OPMD /A A¥OPMD¥BOS

のようにしてOPMDに組み込みBASICからロードしてください。なお，870行のSd="y2, 16"はSd="y2, 19"に変更したほうがよいでしょう。

M1で鳴らす場合には，

A>OPMD /M A¥OPMD¥KAEND

として，M1用のコンフィギュレーションファイルでOPMDを立ち上げてからRUNです。4月号のバーニングフォースのときと同様に，あらかじめM1側の電源を入れ，シーケンサモードにしておく必要があります。

MT-32の場合は楽器側の電源を入れたあと，

A>OPMD /M A:¥OPMD¥KEMT

として，MT-32用のコンフィギュレーションファイルでOPMDを立ち上げます。あとは同様にBASICからRUNです。

今回もFM音原とMIDI楽器が同時に鳴っていますのでミキサーなどでミキシングして聞いてください。そういえば，電波新聞社から簡易ミキシングケーブル(1,300円税抜き)が発売されました。ミキサーをお持ちでない方は購入されるといいでしょう。

(西川　善司)

Language

X68000対応GNU Cコンパイラ

GCC Ver.1.36.01

UNIXでは非常に有名なGNUCコンパイラをHuman68kに移植したものです。Oh!Xでも何度か話題になっていますからご存じの方も多いでしょう。残念ながら最新版ではありませんが，それでも桁外れの高性能を示します。標準的な開発ツールとして使ってください。

Nakamori Akira
中森 章

GNU CC（略してGCC）はANSI規格準拠の高性能なCコンパイラとして有名です。RISC用のCコンパイラとまではいきませんが，CISC用でGCCに勝る最適化を行うCコンパイラはまずないと思われます。

GCCはUNIXのコンパチ環境を作るというGNUプロジェクトの核となるコンパイラで，本来はワークステーションを対象としたものです。しかし，その高性能のためにパソコン（32ビットプロセッサであることが必要）用にも移植されてきています。

今回お届けするのはGCCのバージョン1.36（最新バージョンは1.37.1）のX68000への移植版です。今回のGCCは最新版より2つ前のバージョンですが大きな変更点はないと思われます。

実際，GCCの性能はバージョン1.34あたりからそれほど変わっていません。これではつまらないので，今回の配布に当たってはバージョン1.36の別バージョンを作ってみました（バージョン1.36.01としましょう）。これはオリジナルのバージョン1.36にGMICRO（TRONチップ！）用に作成された乗算ルーチンのパッチ（JUNETより）を当てて作成したものです。

X68000用へのパッチは，基本的には近藤真己氏の作成したバージョン1.31用のものを流用させてもらいました。このバージョン1.36.01の特徴は以下の3点です。

●整数の定数倍はライブラリ関数（__mulsi3)を呼ばずに，シフト，ADD，SUBなどから計算するので乗算が高速に行える。
●BCでコンバートしたファイルをコンパイルできるように2進定数(0bXXXXの形式)を使えるようにした。
●－mc68020オプションで68020用のコードを出力できる（ありがたみはないが)。

参考までにドライストンベンチマーク(Ver.2.1)の結果（OPMドライバなしの状態）は，

現行の1.36	1428ドライストン
今回の1.36.01	1538ドライストン

となっています（コンパイル時のオプションは‘－O －fomit－frame－pointer －fstrength－reduce’)。

ところで，X68000用のGCCではコンパイル時に－finline－functionsを指定するとドライストンベンチマークのコード生成を誤っていましたが，このバージョン1.36.01では誤りません。これはバグが取れたわけではなく，定数倍の乗算でライブラリ関数を呼ばなくなったことに起因します。

なお，私が改造を加えたのはGCCの本体(GCC_CC1.X）のみで，その他（GCC.XやGCC_CPP.X)にはなんら手を加えず，手元にあったバージョン1.31のドライバと近藤真己氏の移植によるバージョン1.37.1のプリプロセッサを収録しています。

また，ネットで出回っているGCCのバージョン1.36には64ビット整数（long long int型）用のライブラリがついていなかった(私の入手当時)ため，ライブラリをアセンブリ言語で自作してみました。それ以外のライブラリに関しては電脳倶楽部で発表されたものと同程度の性能でしたのでオリジナルのものを使用しています。

表1

-o file
GCCによって出力される実行形式ファイルのファイル名をfile.Xとする。
-c
GCCの実行をリンク直前でやめる。このとき入力ファイルの拡張子である‘.c’または‘.s’を‘.o’で置き換えた名前のファイルができる。
-S
GCCの実行をアセンブル直前でやめる。このとき，入力ファイルの拡張子である‘.c’を‘.s’で置き換えた名前のファイルができる。
-E
プリプロセッサのみを実行する。プリプロセッサの出力は標準出力に出力される。
-v
GCCの実行内容を表示する。プリプロセッサやコンパイラのバージョンもプリントされる。
-ansi
GCCの文法をANSIコンパチにする。すなわちGCCでの拡張機能を禁止する。
-traditional
伝統的（UNIXコンパチ？）なCコンパイラの文法をサポートしようとする。
-O
最適化を行う。当然，コンパイル時間は多くかかる，はずなのだが，場合によっては最適化を指定するほうが速いこともある。最適化されて中間ファイルが小さくなった分，入出力に要する時間が有利なのかもしれない。
-Wall
すべてのワーニングメッセージを出す。
-m68020
MC68020用のコードを出力する。
-f flag
CPUに依存しない最適化を指定する。flagとしてはstrength-reduce，omit-frame-pointer，inline-functionsなどがある。これらについては1990年1月号『GCCに見る最適化』を参照。
-Idir
インクルードファイルをdirというディレクトリから探す。
-Dmacro
macroというマクロを値‘1’として定義する。
-Dmacro=defn
macroというマクロを値defnとして定義する。
-Umacro
macroというマクロの定義を取り消す。
-trigraphs
トライグラフを許可する。これは-ansiオプションに含まれている。

使用上の注意

まず，GCC.LZHを展開してGNU.MEのドキュメントを読んでください。GNUはUNIXを中心に活動している著作権を放棄していないフリーウェア群です。GNUの主旨に賛同する人は誰でもこれらのツールを使用することができます。

ディスクに入っているGCCはコンパイラ本体とその周辺のみですので，実際の使用にはXCが必要です（正確にはXCのライブラリとアセンブラ/リンカなど)。XCが実行できる環境CC.Xと同じディレクトリにGCC_CC1.XとGCC_CPP.Xを置いてGNULIB.AをLIBのディレクトリに入れてください。

ただし，プログラムサイズが非常に大きくなっていますので，メモリ1Mバイトの人はほとんど使えないでしょう。増設して

ください。

実行時にスタック関係のオプション指定のエラーが発生することがありますが，無害ですので無視してかまいません。また，BASICコンパイラとして使用する場合，GBC.BATを使ってみるとよいでしょう。漢字を含んだプログラムの異状は knj2oct を通してみてください。

コンパイル時のオプション

コンパイル時のオプションはUNIXとほぼコンパチです。GCCを起動すると，通常はプリプロセッサ（GCC_CPP.X），コンパイラ（GCC_CC1.X），アセンブラ（AS.X），リンカ（LK.X）が順次呼び出されて実行形式のプログラムが作られます。すなわち，拡張子が‘.c’であるファイルはC言語のプログラムとみなされプリプロセッサ，コンパイラ，アセンブラ，リンカが起動されます。また，拡張子が‘.s’であるファイルはアセンブリ言語のプログラムとみなされアセンブラ以降が起動されます。最後に拡張子が‘.o’であるファイルはオブジェクトファイルとみなされリンカが起動されます。

表1にGCCのコンパイル時のオプションの主なものを示しておきましょう。

では皆さん，GCCを活用してください。

［注意］

このプログラム（GCCコンパイラ）の著作権はリチャード・ストールマンおよびFSF(フリー・ソフトウェア・ファウンデーション）に帰属します。このプログラムを使用して発生した損害について，FSFおよび移植者，Oh!X編集室は一切の責任を負いませんので注意してください。また，今後のサポートに対する保証もいたしかねます。

このプログラムを他人に配布することは自由ですが，再配布は必ず無償で行わなければなりません（実費程度の手数料は可）。再配布の際は出所および連絡先などが明らかになるよう心掛けてください。

また，再配布を妨げるような変更を加えることは固くお断りします。

このプログラムを利用するにはC compiler PRO-68Kが必要です。

このプログラムはC compiler PRO-68Kのルーチンなどを使用して作成されています。さらに，このプログラムで作成されるオブジェクトにはC compiler PRO-68Kのライブラリが含まれることになります。使用に対するロイヤリティなどは無償ですが商業目的などで使用する場合は，XCユーザーズマニュアルにあるとおり，XCのライブラリを使用したことを明記してください。

どうしてもこのプログラムのソースプログラムが必要という場合にはOh!X編集室まで連絡してください。

INTEGRAL X1 MAKE FILE 要項

「ついにOh!Xでもディスクサービスやるんですか？　うれしいですね。でも，ちょっと待ってください。X68000とX1じゃディスクのフォーマットが違うじゃないですか！　まさか，X1を見捨てるつもりじゃないでしょうね」

というやりとりの数分後，「INTEGRAL X1」プログラムは，ディスクにのることになりました。プログラムの内容は本文を見てもらうとして，ここでは，付録のディスクから，いかにしてX1のファイルを作り出すか？　について説明します。

とりあえず，先に注意点をあげておきます。

・付録ディスクからX1用ファイルを作り出すには，2HDドライブが2台必要。

・2HDドライブを持たないX1turboでは，直接ファイルを作ることはできない。しかし，作成したプログラムを使うことはできる（友達などにたかりましょう）。

・ノーマルX1では作成したプログラムを使うことはできない（今月はできないけど，ちゃんと連載中にサポートするつもり）。

それでは手順を説明します。

1）BASICのCZ-8FB02を使って135ページのリスト1を入力して間違いがないかどうか確認したら，SAVEしておいてください。

2）フォーマット（X1標準2HD）したばかりの，まっさらな2HDディスクを用意します。

3）135ページのリスト1をRUNさせて，ドライブ0:に付録のディスク，ドライブ1:に2）で用意したディスクをセットします。用意ができたらYキーを押してください。

4）「読んで書いて」を20回ほど，エラーなく実行できれば，ドライブ1:のFILESを表示します。次の3つのファイルができていることを確認してください。

INTEGRAL.X
FDC　　.OBJ
COMMAND.X1

3つできていれば成功です。あとは次の実行方法をよく読んでください。もしエラーが出たら，2)のFORMATのところからやり直してください。

ファイルを作ったら，「INTEGRAL.X」をRUNしてください。自動的に2つのファイルをロード・実行します。以後，BASICを立ち上げたときとか，別のプログラムのあとに実行する場合は，「INTEGRAL.X」を使ってください。STARTUPに登録しておけば便利でしょう。また，バンクメモリを使わないのであれば，CZ-8FB03(Z-BASIC）でも動かすことができます。

KAME-DOSでは，テープ・2DD・ハードディスク・8インチなどのサポートはしていません。ひとつのディレクトリで扱えるのは，128ファイルまでです。

このプログラムの初期設定は，Zの2HDディスクに合わせてあります。もし，読み込めないことがありましたら，INTEGRAL.Xファイル中，1150行の“&H10”をもう少し大きくしてみてください。また，ステップレートが3ms以外のドライブを使っている方は1120行の’を取って1130行に’をつけてください。　（鶴）

参考：祝　一平，『試験に出るX1』

その他のファイル

●FLOAT2+/3+.X

SYS.LZHに圧縮されているFLOAT2+.X, FLOAT3+.Xは以前Oh!X誌上で発表された高速数値演算ドライバです（FLOAT2+.X：1988年8月号山口正氏作，FLOAT3+.X：1989年3月号長井清氏作）。

使用するときはCONFIG.SYSで，

DEVICE＝FLOAT3+.X

のように指定します。高速ドライバとなると気になるのは動作速度ですね。レイトレーシングソフトなどの演算を多用されるアプリケーションで見る限り，速度はそれぞれ4割増し（実行時間が6割）くらいとなります。

誠に申し訳ない話ですが，FLOAT2+.Xにまだ型変換関係(?)のバグがあるらしく，今月掲載のANGELを始め，一部のアプリケーションで計算間違いを起こします。どなたか1989年3月号のソースプログラムを参考にデバッグしていただけるとうれしいんですが。

●CARD.FNC

1990年5月号で発表した毛内俊行氏によるカードゲーム支援システムCARD.FNCを生成するためのファイルとサンプルプログラムです。

CARD.FNCの作成手順は以下のとおり。まず，GAME.LZHを解凍します。MAKE.BASを実行すればFONT.DATを展開してCARD.FNCができあがります。あとはこのCARD.FNCをBASICの入ったディレクトリに運んで，BASIC.CNFに，

FUNC＝CARD

の1行を加えるだけです。

●SWORD_2D.D

X68000, PC-286版“SWORD”で使用できるサンプルです。最初におことわりしておきますが，すべてが完動するとは限りません。プログラムの使い方は各バックナンバーを参照してください，といっても，まったくわからない人もいるでしょうからエディタアセンブラREDAの最低限の使用法のみ解説しておきましょう。

#L　REDA
#J3000

でREDAが起動します。最初はアセンブラモードですから，

E

でエディタに移ります。さらに，

E

でエディットモードに入ります。

行番号直後のスペースはコントロールカラムと呼ばれ，ここに入力された1文字はコントロールコードのように機能します（S-OSにはコントロールキーはありません）。たとえば，

+	10行空白を挿入
−	空白行を削除
[	上1/2ページスクロール
]	下1/2ページスクロール

などです。エディタからはQでアセンブラに戻り，Aでアセンブルします。

ウイルス検出プログラム

DOCTOR.R

残念なことにX68000上のウイルスが一部のユーザーの間に広まっており，とうとう市販のソフトウェアにまで混入するという事態が起きてしまいました。今回はこの種のウイルスに対応するワクチンを付録のディスクに収めてありますので，予防の意味も含めてぜひともチェックをお願いします。

編集室

4月24日，X68000用ウイルスが市販ソフトに入っていたことが新聞で報じられました。アートディンクの新作，ファーサイドムーンのディスクの1枚にウイルスが混入していたというものです。

ウイルスが混入していたのが立ち上げに使うシステムディスクではなかったのが不幸中の幸いとはいえ，誤操作をするとウイルスがX68000のSRAMに住みついてあらゆるディスクを汚染してしまいます。

このウイルスは内蔵タイマが1990年7月以降になると起動しようとして挿入されたディスクの内容を破壊してしまいます。そのほか，感染時にOPT.1キーを押さずにプロテクトシールの貼っていない市販ソフト(の一部)を起動すると，以後そのディスクから起動できなくなる場合があります。

FAR SIDE MOONは4月13日に発売されましたが，ウイルスはまもなく発見されたようです。16日，編集部でも発見者のひとりから連絡を受け、すぐさまアートディンクでも確認がなされました。このため，アートディンクでは商品の回収を決め，すでに購入してしまったユーザーに対してはワクチンを配布するためユーザー登録を呼び掛けることになったのです。

しかし，このウイルスは以前から広まっているもののようです。なんとか7月までにこのウイルスを根絶するため，読者の皆さんに協力をお願いしなければなりません。

今回付録のディスクに収録されているワクチンはアートディンクに提供したものと同じです。ワクチンは決して完全なものではありませんが，使用方法を理解して正しく利用すれば大切なマシンとフロッピーをウイルスの感染による被害から守ることができるでしょう。

このワクチン“DOCTOR.Rには3つの機能があります。

1) ウイルスに感染したX68000のSRAMを初期化する(zap.x)。

2) SRAMにワクチンを常駐させてウイルスの侵入をチェックする(tfr2.x)。

3) 感染したディスクを治療する。

まず読者の皆さんにお願いしたいのは，1)と2)です。SRAMを独自の用途に使われている方以外はぜひともzap.xとtfr2.xを実行してください。そうすれば，ウイルスに感染したディスクで起動しようとする際に，警告を発してすぐさま治療モードに入ります。詳しくは下の「ワクチンの使い方」を読んで自分のディスクをチェックしてください。

今回のウイルス

ウイルスをダンプすると“Initial Program Loader”という文字と“X68k Force”という作者名らしきものが見えます。現在，バージョン1.00，1.02，1.05というものが確認されています。これらのウイルスはフロッピーディスクのIPLからSRAMに侵入し，ほかのフロッピーディスクのIPLを汚染していきます。HDDについてはちゃんとチェックしているので通常の動作では安全なようです。

Oh!Xでは昨年末，編集部のX68000に入っていたウイルスチェッカ（昨年6月号で発表した旧doctor.r)が何度か破壊されていることに気づきました。調査の結果，それが新たなウイルスのせいであるものと判明しました。発見後，ただちにこれを解析し，ワクチンの原形となるものを作成しました。その後，同種のウイルスに関するものと思われる情報がいくつか入り，本誌では再びこの問題を取り上げ，新たなワクチンを発表する必要があると判断したわけです。そのため，これまでのチェックに加えて，せめて標準のIPLかどうかチェックする機能がないとウイルスを見過ごしやすいということで，中森氏にIPL部分のチェックをもう少し徹底するように依頼してDOCTOR.Rの現バージョンとなりました。

プログラムのチェックが終わり，当初の予定では4月号で満開製作所と連動して掲載されるはずでしたが，今回のディスク配布の企画が持ち上がり，より確実な対処法を練っているうちに大きな事件が起こってしまいました。結果として対応が遅れてしまったことをお詫びします。

ワクチンの使い方

付属ディスクのワクチンDOCTOR.Rは圧縮されていますので，使用の前に解凍(展開)という作業が必要です。できればワクチン専用のディスクを作るのがよいでしょう。コピーしたシステムディスクをAドライブに入れてCOMMAND.Xを起動(戦車のアイコンをダブルクリック)してください。付録ディスクをBドライブに入れ，に続いて，

```
A>B:LH -E B:DOCTOR⏎
```

と入力してください。これでAドライブのシステムディスクにDOCTOR.Rが展開されます。

さっそく，SRAMを初期化してDOCTOR.Rを組み込みましょう。

```
A>zap⏎
A>tfr2⏎
```

と入力すれば組み込み完了となります。

それでは，ディスクがウイルスに冒されていないかチェックしてみましょう。ここでリセットしてください。「ドライブ0のHumanは正常です」というメッセージが出ればOKです。

もし，あなたのX68000が感染していたなら，ここまでの処理で、作成したシステムディスクも感染し，「このディスクはウイルスに冒されています」と表示されるかもしれません。その場合はリターンキーで治療されますが，その際，ライトプロテクトシールが貼られていないことを必ず確認してから治療を行ってください。

また、DOCTOR.Rが治療用に持っているIPLと異なる場合は，ドライブ1にマスターディスクを入れてリターキーを押してください。

以上で，治療が完了しました。以後，ワクチンの入ったシステムディスクはプロテクトシールを貼って保存してください。

さて，ワクチンを組み込んだあとは，立ち上げのたびにメッセージ表示を確認するよう心掛けてください。万一，ウイルスがSRAMに侵入した場合にはワクチンが壊され，メッセージが出なくなります。その場合には先の手順を繰り返してください。

また，SRAMが感染するとディスクから起動できなくなることがありますが，その場合にはOPT.1を押しながら立ち上げてください。

X68000はOPT.1キーを押しながらリセットすることで必ずフロッピーディスクから起動できるように設計されています。ですから，一部新聞で報道されたようにディスクから立ち上げ不可能になることはありえません。困ったときは一度コンセントを抜き落ち着いてOPT.1を押してください。

doctorプログラムの概要

マニュアル代わりにdoctorプログラムの動作について説明しておきましょう。

doctor. rが起動されるとセルフチェックを行ったのち，ドライブ0に挿入されたディスクのIPLのチェックを開始します。もし，セルフチェックが失敗すればプログラムの実行は中断します。この場合はdoctor.rが書き換えられている可能性がありますから，あらかじめ安全な場所に保存しておいたdoctor. rのバックアップをコピーしてきて再実行してください。

次にIPLがシャープ/ハドソンの純正品と一致するか否かをチェックします。純正のIPLと完全に一致する場合は実行は終了します。もし純正のIPLの一部分だけが書き換えられている場合は，なんらかのウイルスに感染している可能性が高いので治療モードに入ります。

IPLが純正でない場合(いくつかの市販ソフトやある種のPDS)は前回のウイルスと今回の新種のウイルスのチェックを行います。もしどちらかのウイルスに感染している場合は治療モードに入ります。しかし，ここでのウイルスのチェックに引っ掛からないときは単にメッセージを表示するのみでなにも行いません。この場合はウイルスに感染しているか否かの判断ができませんから注意が必要です。また，このdoctor. rはOS-9のディスクに関してはまったく無力なので悪しからず。

さて，治療モードではドライブ1に挿入されたディスクのIPLをドライブ0のディスクに書き写すことで治療を行います。当然，ドライブ0のディスクにプロテクトシールが貼られていてはなりません。そして，このときドライブ1のディスクもウイルスに感染したディスクであればディスクを吐き出して新たなディスクの挿入を待ちます。

ここまでは旧doctor. rとコンパチな機能です。ただし，今回のdoctor. rではドライブ1にディスクを入れずにリターンキーのみを押せば純正のIPLで(バージョンが判明するときは正しいバージョンで，そうでないときはバージョン2.0で)治療を行う機能を追加しています。ただし，この新機能を純正のIPLを持たないゲームやPDSのディスクの治療に使用すると悲惨なことになりますから注意しましょう。

また，エラーチェックはかなり甘めです。プロテクトシールが貼られていても治療終了のメッセージを出して起動するほか，ディスクがささっていなかったり，エラーが発生したときにはこのチェッカをすり抜けて起動することがあるので気をつけてください。ウイルスがすり抜けて起動するとSRAMが汚染されチェッカを破壊してしまいますので十分な注意が必要です。

doctorプログラムの使用法

doctorプログラムは単体でも実行可能ですが，SRAMに常駐させてドライブ0からディスクを立ち上げる前にチェックを行うようにするのがいいでしょう。そのためのプログラムとしてzap. xとtfr2. xが付属しています。zap. xはSRAMを初期化するプログラムです。すでにSRAMがなんらかの目的で使用されている場合はzap. xの実行によりすべてが無効になってしまいますので心して使ってください。

tfr2. xはカレントディレクトリのdoctor. rをSRAMに転送し，起動時に自動的にディスクをチェックするようにさせる転送プログラムです。これはOh!X1989年6月号に掲載されたtfr. xと同じものですが，doctor. rのサイズがいくらであっても対応できるように改造してあります(オリジナルは2000バイトまでしか転送できない)。

tfr2. xを起動するとdoctor. rがSRAMに転送され，SRAMの使用状況はProgram，BOOTはRAM0(&HED0100)になるようにメモリスイッチが変更されます。これにより，X68000の起動後はまずドライブ0の状況をチェックし，異常がなければSTDのモード(ドライブ0，ドライブ1，ハードディスクの優先順位)でブートすることになります。ただし，この場合SRAMをRAMディスクなどの用途で使用することはできなくなります。基本的な使用法は本誌1989年6月号(pp. 92-96)も参照してください。

SRAMというのは，いつのまにか書き換えられたり破壊されることがあり，プログラムやデータの一部が破壊されると誤動作が発生します。安全のため定期的にzap,tfr2を行ってください。

なお，このプログラムの配布は自由ですが，必要以上の改造を加えることは禁止します。また，操作の誤りまたはこのプログラムの類似品によりなんらかの被害が発生してもOh！X編集部は一切責任を持てません。

*　　*　　*

市販ソフトへのウイルス混入。いつかは起きるかもしれないと誰もが恐れていた。

一般のユーザーの方でも漠然とした予感はあったことと思います。

もちろんマスターアップの際には，どのソフトハウスでも細心の注意を払っています。今回はそれまで確認されていなかったウイルスが急に侵入してきたこと，および「起動用でないディスクのIPL部分」というのが盲点となってしまったようです。

ウイルスというものの性格からして，仮に現状に比べ2重3重にチェックシステムを強化しても，やはりいつかはこのような事態が発生したかもしれません。

ウイルスそのものの被害はもちろんですが，多くの人は無責任な報道による社会的影響を危惧していたのではないでしょうか。これまでに発生したコンピュータウイルス関連の報道を見る限り大手新聞紙面でさえ，一般の人にはいたずらに不安をあおるだけではないかと思えます。

当事者となったアートディンクはもちろん，その他ソフトハウスやショップ，流通関係が慎重な対応に苦慮し，ソフト回収とウイルス対策に奔走していた直後の新聞報道でした。結果的に無神経な報道発表となってしまったことが残念です。

X68000ユーザーの皆様へ

すでに，各報道でご承知のことと存じますが，4月13日に発売した初期出荷バージョン「ファーサイドムーンX68000版」(機種表示部分が真っ赤なもの）の一部にコンピュータウイルスが発見され，X68000ユーザーの皆様のみならず，社会全体をおさわがせしましたことに深くお詫び申し上げます。

問題となっているウイルスは，X68000本体のSRAMに付着する「NX68K IPL V1.02」という種類のもので，症状は今年の7月になったらドライブに差し込まれたディスクのデータを破壊するかもしれないという極めて悪質なものです。

しかし，このウイルスを撲滅することができる「ワクチン」がOh！X誌のご協力をいただきまして，すでに準備されており（このワクチンは本号の特別付録のワクチン＝DOCTOR.Rと同じものです）ただいま，弊社登録ユーザーの皆様に無料で配布いたしております。まだユーザー登録をされていないお客様は，大至急，当社まで「ユーザー登録ハガキ」をご返送ください。

未来あるパソコンライフ，安心してコンピュータに親しむことができる社会環境のためにも，ユーザーの皆様方の絶大なるご支援をお願いする次第です。

何卒，ご理解ご協力をたまわりますよう，よろしくお願い申し上げます。

平成2年5月　株式会社アートディンク

付録ディスクに収められたdoctor.r，tfr2.x，zap.x の各プログラムはフリーウェアとしますが，配布の際には必ずこのページ（56～57pp.）のコピーを添付してください。

Tools

インテリジェントソースジェネレータ

DIS.X

Abe Kouta
安倍 広多

よい子に贈る魔法のプログラム。あらゆるオブジェクトからソースプログラムを作り出してしまう，それがDIS.Xです。技は盗み盗まれて磨かれていくものです。DIS.Xは先人のプログラミングノウハウを学んでいくには最高のツールといえるでしょう。有意義に活用してください。

えーっと。お久しぶりです。はい。遥か以前にX1用の拡張漢字BASICを発表させていただいた安倍と申します。あれから？年，私のマシンもいつの間にやらX1からX68000へと変わり，月日の流れを感じる今日この頃です。

今回はOh!Xに付録ディスクがつくということで，以前よりこつこつと作ってきたソースコードジェネレータDISを発表できることになりました。解析好きの皆さん，ぜひ一度使ってみてください。

制作の動機

ここで発表するソースコードジェネレータDISとはその名のとおり，Human68kの実行ファイルから比較的まともなアセンブラソースをフルオートで生成するものです。生成されたソースはそのままアセンブラas.xでアセンブルすることができ，できた実行ファイルは元の実行ファイルとほぼ同一になります。コマンドや市販ソフトの解析などに非常に有効です。

ソースコードジェネレータを作ろうと思ったのは，大学に入ってX68000を買ってすぐのことです。私の大学のパソコンでは2HDフォーマットを読めず，2HC(注1)しかサポートされていませんでした。Humanで2HCディスクを読むためにはデバイスドライバを書かねばなりません。

そこで68000のアセンブラを勉強し，ない資料を集めてデバイスドライバを書いたのですが，どうもうまく動きません。そうです。Human68k ver.1は，1024バイト/セクタ以外のブロック型デバイスをサポートしていなかったのでした（ver.2は大丈夫のようですが)。2HCのドライバは必要不可欠だったので，しかたなしにHumanを逆アセンブルして解析し，1024バイト/セクタ以外もサポートするようパッチを当てることに成功しました。そのときHumanのソースがあればもっと楽に解析できるのにと思ったのがソースコードジェネレータを作り始めたそもそもの動機です。

DISの特徴

DISは私の所属するDōGAでは，「とんでもない」「悪魔のような」ツールと呼ばれていまして，それにふさわしく(?)次のような特徴を備えています。文字列の自動認識などはなかなか気持ちのよいものです。

- .x .r .zファイル，および単なるバイナリファイル(注2)をサポート
- .text .data .bssのセクションを分離する
- 文字列を自動的に認識する
- もし存在すれば，.xファイル中のシンボルネームを使う
- .xファイルに含まれている再配置情報も利用する
- 命令の中を指すラベル(注3)もサポート
- DOSコール，IOCSコール，浮動小数点演算命令（FEcall）もサポート
- リラティブオフセットテーブル(注4)の自動認識
- 基本的にはフルオートですが，おかしいところがあればユーザーが指定することも可能

使い方

doscall.mac，iocscall. mac，fefunc.hを読み込むので，環境変数includeにこれらの格納されているディレクトリ名をセットしておいてください。

実行はコマンドラインより，

dis [options] 実行ファイル名 出力ファイル名 [options]

です。

基本的にはこれだけでアセンブラソースを生成しますが，より細かい設定をするために表1のような数多くのオプションが用意されています。本文ではスペースの都合でオプションの説明などを少々割愛しましたので，詳しい説明は付属のreadme. docを参照してください。

アルゴリズム

不毛なプログラムだけにすべて解説するのはとても大変なので，概略のみ説明します。別にわからなくても十分使えます。

DISが実行ファイルを解析する際にいろいろなメッセージを表示します。まずそれらを簡単に説明しましょう。先頭の数字は説明の都合上つけたフェーズ番号です。

1) プログラム領域解析中です

プログラムの実行開始アドレスより通る可能性のあるところをしらみつぶしに解析します。ただし，アドレステーブルやレジスタ間接ジャンプなどの原因で，このフェーズで解析できなかった中にもプログラム領域は存在するはずです。

2) データ領域解析中です

前のフェーズでデータ領域として残ったところのうち，アドレスに依存している(注5）ところをラベルバッファへ登録します。

3) アドレステーブルから捜しています

連続して3つ以上続いているアドレスに依存したところを，アドレステーブルと見なして，そこを手掛かりに解析を試みます。

4) データ領域の中からプログラム領域を捜しています

データラベル中でサイズが不明なものすべてから，プログラムと見なして解析を試みます。

5) 文字列を探しています

データラベル中でサイズが不明なものすべてを，文字列と見なせないかチェックします。

6) ラベルチェック中

主に命令の中を指すラベルを探します。

7) ソースを生成中です

作りあげたラベルバッファを参照してソースを出力します。第1フェーズは下で説明する関数analyzeをコールして，実行開始アドレスよりひととおり解析します。

関数analyze(pc)はDISの心臓部とでも呼べるものです。この関数はpcから逆アセン

ブルして，プログラム中を指すラベルを登録していきます。また，分岐命令などを見つけると自分自身を再帰呼び出しすることによってpcから通る可能性のあるところをすべてトレースし，解析します(図1参照)。ただしjmp(a0)などはあきらめてしまいます。

また，本当はデータであるところを誤ってプログラムと見なさないようにするため，以下のような場合はプログラムとして見なしません。

・pcが奇数アドレス
・pcがtextセクションを外れた
・命令がアドレスに依存している
・pcがori.b ＃0，d0から始まる
・ori.b ＃??，d0が2つ連続している
・オペランドが絶対ロングアドレスやロングワードのイミディエイトデータでないのにアドレスに依存している
・未定義命令(注6)
・分岐先が奇数アドレス

分岐先で以上のような条件に引っ掛かって解析中に上のような条件でそこがプログラムでないと判断した場合，プログラムでないと判断するまでに登録してしまったラベルを取り消します。取り消さないと，妙なラベルがいっぱいできてしまいます(図2参照)。

第2，第3フェーズではデータ領域中のアドレスに依存したところに注目して処理しています。たいしたことはしていません。

第4フェーズはDISの解析でも特徴的なところです。このフェーズではサイズが不明のすべてのデータ領域の先頭からとりあえず関数analyzeを使って解析してみて，おかしくなければプログラム領域と思い込みます。analyzeのチェックはかなり厳しいのでデータ領域をプログラム領域と間違えることはそんなにないはずです。

第4フェーズを繰り返して，もうデータ領域中からプログラム領域と見なせるものがなくなったら，第5フェーズに移ります。第5フェーズでは，文字列を自動的に判別します。アルゴリズムは単純で，サイズが不明のすべてのデータ領域の先頭から表示可能な文字および一部のコントロールコードが続いているか，漢字の第1バイトの後ろに第2バイトとして正当なものがきているかをチェックしているだけです。これだけのことでかなりの確率で文字列が判別できるようです。

第6フェーズ。主に以下の処理をします。

図1 analyze( )の再帰による解析の様子1

```
top
        .....
        bsr    sub1
        .....
        rts
sub1
        .....
        bcc   sub2
        .....
sub2
        .....
        rts
```

上のようなソースの場合、下のように解析していきます。

```
                          ↓解析中の文字です
analyze( top )          > 実行開始アドレスより解析開始
analyze( sub1 )         > bsr で sub1 より解析
analyze( sub2 )         > bcc で sub2 より解析
return( TRUE )          < リターン命令で戻る
return( TRUE )          < 解析済みのアドレス( sub2 )に達し
                          たので戻る
return( TRUE )          < リターン命令で戻る
```

注1　2HC
512バイト/セクタ，15セクタ/トラックというMS-DOSフォーマットのひとつ。

注2　単なるバイナリファイル
ファイルにヘッダ部分がなく，ファイルの先頭から68000のアセンブリコードがベタに書いてあるファイルのこと。IOCS ROMをファイルへ書き出したりしたものなどが該当します。

注3　命令の中を指すラベル
命令の中を指すラベルとは下のようなものです。

```
label:  move.w  #$0000, d0
        ......
        move.w  d1, label+2
```

命令のオペランド部を書き換えるようなところもきちんとラベルを振ります。また，この文中でのラベルとは基本的にアセンブラでのラベルと一緒と思ってもらって結構です。

注4　リラティブオフセットテーブル
リロケータブルなプログラムでテーブルを作るときによく用いられます。テーブルにはテーブルの先頭番地からのオフセットが符号付き1ワードで書かれます。分岐テーブルによく用いられますが，データテーブルにももちろん使用可能です。
(例)

```
    lsl.w  #1, d0                    * 2倍
    move.w table(pc, d0.w), d0       * ←
    jmp table(pc, d0.w)              * ←
table
    dc.w   label1-table
    dc.w   label2-table
    dc.w   label3-table
    dc.w   label4-table
```

DISのリラティブオフセットテーブルの自動認識は，上の←の命令シーケンスを判別しているにすぎません。なお“リラティブオフセットテーブル”の用語は参考文献3より借りてきたものです。

注5　アドレスに依存している
.xファイルは主記憶のどこにロードされても動くように，プログラム中のどこがロードされたアドレスに依存しているかを記憶しています。
たとえば，

```
label:  lea.l  label, a0
```

の場合，lea.lのソースオペランド(1ロングワード)がアドレスに依存しています。

注6　未定義命令
68000には豊富な未定義命令(?)があり，DISの解析に役立っています。DISで本来の未定義命令以外に未定義と見なすのはデフォルトで次のようなものです。
・実行時にアドレスエラーの起こるであろう命令
・A line trapおよびDOS call，浮動小数点emulatorで使用されていないF line trap
・バイト操作命令の不定バイトが00_H，FF_Hのいずれでもない
・ビット番号をイミディエイト値で与えるビット操作命令の第3バイトが0でない
・movemでレジスタフィールドが0(どのレジスタも転送しない)

注7　不定バイト
オペレーションサイズがバイト(.b)の命令で，かつソースオペランドがイミディエイトデータの場合，命令の第3バイトは定義されていません(不定)。しかし，as.xを使っている限りこの第3バイトは00_HかFF_Hのどちらかのようなので，通常DISはここをチェックして00_HかFF_Hのどちらでもなければ未定義命令ということにしてしまいます。

基本的には第7フェーズのための前処理のようなものです。

・命令の中を指すラベルを探し出す。

・プログラム領域としているところに未定義命令がないか調べ，あれば警告を出力する。ラベルファイルの書き間違いなどで発生。

第7フェーズではソースを出力します。そんなに特殊なことはしていません。move q.l ＃??，d0の後ろがtrap ＃15だったら，IOCSコールとして処理しています。

DIS のアルゴリズムの概略は以上ですが，「アルゴリズム＋データ構造＝プログラム」(だったっけ？)ですので，DISでもっとも重要なデータであるラベルの管理について少し説明します。

DIS はラベルの記憶のために以下のような構造体のテーブル(ラベルバッファ)を使用しています。

・アドレス

・属性

・このアドレスが参照された回数（ラベルの取り消しのために用いる）

属性としては以下のものがあります。どのような場合にその属性として登録されるのかを示します。

・プログラム

分岐命令での飛び先などとなっている場合

・データ

・バイトデータ

move. b　label, d0などの場合

・ワードデータ

move. w　label, d0などの場合

・ロングワードデータ

move. l　label, d0などの場合

・不明

lea. l　label, a0やmove. l ＃label, a0などの場合

・文字列

「不明」のうち，文字列として認識したもの

・リロケータブルオフセットテーブル

リロケータブルオフセットテーブルを使うときの命令シーケンスを自動認識した場合

/eオプションで出力されるラベルファイルはDISが解析を終えた状態のラベルバッファをテキストファイルの形式で出力したものです。これに変更を加えて/gオプションでラベルファイルを読み込ませれば，ユーザーの指定したようなソースを出力することができます。詳しくはreadme. doc を参照してください。

図2　analyze( ) の再帰による解析の様子2

```
top

     .....
     lea.l    sub1.a0
     .....
     rts
sub1      * 本来はデータのところ
     .....
     bcc      sub2
     .....
sub2

     .....
     undefined instruction
```

上のようなソースの場合、下のように解析していきます。

```
                    ↓解析中の文字です
analyze( top )      > 実行開始アドレスより解析開始
return( TRUE )      < リターン命令で戻る
analyze( sub1 )     > sub1 のサイズが不明なのでプログラム
ではないかと解析してみる
analyze( sub2 )     > bcc で sub2 より解析
return( FALSE )     ? 未定義命令で戻る
                      sub2 からはデータでした。
                      sub2 から登録してしまったラベルを取り
                      消す
return( FALSE )     ? 分岐先がおかしかったので戻る
                      sub1 からはデータでした
                      sub1 から登録してしまったラベルを取り
                      消す( sub2 というラベルも )
```

注意

DIS の生成したソースファイルの著作権は，元の実行ファイルの著作者にあると思われるので取り扱いには十分注意してください。

DIS. Xは強力なツールです。使い方をちょっと誤れば，不毛で危険なツールになります。しかし，有意義な目的で使えばこれ以上に頼もしいものもありません。

おわりに

このプログラムは，近藤版GCC 1.36 でコンパイルされました。

締め切り間際まで続出するバグに苦しみましたが，一応発表できるかたちになってよかったよかった。徹夜でデバッグなんか二度とするものか。しかし原稿を書くって大変……。

謝辞　以下の人たちはとんでもない実行ファイルを喰わせてバグをいっぱい見つけてくれました。改めて感謝いたします。

DōGAの遊び人松井，FFE三保

大学にこないネットワーカーたかし(BA90)

また，DōGAのかまたゆたかさんにもいろいろとお世話になりました。感謝いたします。

〈参考文献〉

1. 68000プログラマーズハンドブック，宍倉幸則，技術評論社

2. kaiseki.doc, △そると▽, High Speed Linker

3. セミオートディスアセンブラDSK，川本琢二，アスキー，1989年8月号

4. GNU C Compiler document, Free Software Foundation

文中敬称略させていただきました。

表1　オプションの使い方

ここで挙げたのは，DIS.Xの持つオプション群の一部です。さらに詳しくはディスク内のドキュメントを参照してください。

解析に関するオプション

/h　データ領域中の$4e75(RTSのコード)の次のアドレスに注目する

素直でないコーディングをしたプログラムの場合，実際にはプログラム領域なのに生成ファイル中ではデータとして出力される場合があります。そういうときはこのオプションをつければたいていのところを解析してくれるはずです。

/i　分岐先で未定義命令などを発見しても呼び出し側をデータ領域にしない

通常は「未定義命令のあるところに飛ぶような奴は俺はプログラムと認めねぇ」ですが，このオプションをつけると，「そんなことは俺の知ったことじゃねぇ」になります。ワークエリアにプログラムを書いてそこへジャンプするようなプログラムをソースジェネレートする場合に有効かもしれません（しかしどうやったらそんなことがわかるのだろうか)。

/j　実行時にアドレスエラーの起こるであろう命令を未定義命令と見なさない

通常はmove.l $00000001, d0 などの，実行時にアドレスエラーが起こるであろう命令は未定義命令と見なしますが，このオプションをつけるとそのチェックをしません。

/k　命令の中を指すラベルはないものと見なす

命令の中を指すラベルの存在しないプログラムで，このオプションを指定すると生成ファイルの質がよくなる場合があります。しかし存在するプログラムで指定すると生成ファイルの質がかなり悪くなります。とりあえず，/kオプションをつけずにソースジェネレートしてみておかしいところがあれば，/kオプションをつけてやってみて，2つを比較してみるとよいかもしれません。tree.x を/kオプションあり/なしでソースジェネレートしたものを比較すると違いがわかると思います。

/f　バイト操作命令の不定バイト(注7）のチェックをしない。

(効果その1)

不定バイトチェックをしないようにします。

(効果その2)

不定バイトに対するラベルが存在した場合，通常は「不定バイトを書き換えるような奴はいないよなぁ」と信じてその命令（不定バイトのある）領域をデータ領域にしてしまいますが，/fオプションはこのチェックもしないようにします。このオプションをつけた場合，生成ファイルをアセンブル/リンクしたものが元の実行ファイルとまったく等しくはならないはずです。

/u　A line trapおよびDOS call，浮動小数点 emulator で使用されていないF line trapを未定義命令と見なさない

たまに，A line trap を使っているプログラムがあります(Ma＊ Emulatorや，S＊-Window)。それらをソースジェネレートするときにつけてください。なお，使用されているかどうかの判断は，doscall.mac，fefunc.h にシンボルが存在するかどうかによります。このオプションをつけないと，Human ver.2で拡張されたDOS callはdoscall.macを書き換えない限り未定義命令とされます。

/y 「text セクションの中のデータ領域のうち，サイズが不明なものすべてをとりあえず逆アセンブルしてみて，おかしくなければそこをプログラム領域と思い込む」ことをしない

DIS のデフォルトでは，ひととおり解析したあと，サイズのわからないデータは「プログラムちゃうか？」と疑って，片っ端から解析しようと試みます。/yオプションをつけると，「片っ端から」ではなく，データの終わりがリターン命令，ジャンプ命令などと見受けられるもののみを「プログラムちゃうか？」と疑います。詳しくはアルゴリズムの解説を参照してください。

このオプションをつけると解析の対象となる領域が少なくなる分だけ，解析時間が減りますがプログラム領域をデータ領域として残してしまうことが多くなるでしょう。しかしつけないと，データ領域をプログラム領域としてしまうこともあるかもしれません。

/n num　文字列として判定する最小の長さ(デフォルトは3)

DIS は文字列領域を自動的に判別します。ただ，文字列判別を完璧に行うのは不可能で，文字列でないところも文字列としてしまう場合があります（特に短い文字列)。このオプションを指定すると，それ未満の長さは文字列として判定しません。また，num に0を指定すると，文字列の自動判別を行いません。

生成ファイルに関するオプション

/b　分岐命令のサイズを常に省略する

分岐命令のサイズとはBccなどの .w .sのことです。デフォルトでは，

1)　ショートブランチで済むのにワードブランチをしている場合

2)　アセンブラのバグ（readme.doc参照）を回避する場合

にサイズをつけます。このオプションをつけると 1）の場合でもサイズが出力されません。

/o num　文字列領域の横バイト数（デフォルトは70）

ソースファイル中での文字列領域の横バイト数を指定します。

/w num　データ領域の横バイト数（デフォルトは8）

データ領域の横バイト数を指定します。デフォルトでは，

dc.b　$??, $??, $??, $??, $??, $??, $??, $??

のように横は8バイトです。データの多いプログラムでは16にしたほうがよいかもしれません。

/a[num] num　行ごとにアドレスをコメントで出力

num を省略すると5行ごと。ラベルファイル（後述）を編集するときなどは便利かもしれません。

/r　文字列に16進数のコメントをつける

/x　オペコードに16進数のコメントをつける

```
(例)  ori.b    #xx, d0    * $0000, $00xx
      rts                 * $4e75
```

実行ファイルに関するオプション

/d　デバイスドライバのときに指定

実行ファイルがデバイスドライバのときは必ず指定してください。ほんのわずかに生成ファイルの質がよくなります。デバイスドライバでないときは指定しないでください。妙なことが起こるかもしれません。

/z base exec　(base execは16進数のみ)

指定した実行ファイルを単なるバイナリファイルと見なす。その際，そのファイルはbaseから読み込まれてexecから実行されるものとする。たとえば2HDディスクのIPL部を読み出して解析する場合，dis 2hdipl 2hdipl.s/z 2000 2000などと指定する。ROMのIPL/IOCSを解析する場合，dis rom rom.s /zff0000 ff0010などと指定する（/hをつけたほうがよいかもしれない)。当然IPLなどはdb.xなどでHuman68kのファイルに落としておかねばなりません。

ラベルファイルに関するオプション

/e[filename]　ラベルファイルの出力

ファイル名を指定しなければ，出力ファイルのドライブのカレントディレクトリに拡張子が .LABのファイルを作る。

/g[filename]　ラベルファイルの読み込み

各種バッファ領域に関するオプション

/m num　ラベルバッファの数を指定（デフォルトは4096）

/s num　シンボルバッファの数を指定（デフォルトは1024）

その他のオプション

/q　解析中/出力中のメッセージを出力しない

文字の説明

- ＞　分岐命令などで，解析の深さが深くなった
- ＜　リターン命令などで，解析の深さが浅くなった
- ?　未定義命令などを発見して，解析の深さが浅くなった
- #　データ
- s　文字列
- !　ラベルチェックで，命令の中のラベルを発見した
- *　ラベルチェックで，命令の中のラベルを解消した
- .　ラベルひとつ
- -　そのままではプログラム領域からデータ領域に突入してしまうところを発見/修正した
- +　ラベルチェックでプログラム領域中に未定義命令などを発見した
- r　リラティブオフセットテーブル
- $　BSSラベルひとつ

/v　単なる逆アセンブルリストを出力する

このオプションはおまけです。db.xの出力をリダイレクトするよりは見やすいかな……。

Graphic

対話型CGAシステム

ANGEL

アンチエイリアシングや透明体にも対応したポリゴンタイプのユニークな言語型3Dレンダリングシステムです。Zバッファを圧縮する独自のSバッファアルゴリズムを採用し小規模のシステムでも大量のデータが処理できます。パーソナルCGAの世界を広げてください。

Moriyama Hiroki
森山 弘樹

まずは単刀直入にいわせてください。

「私の希望はただひとつ，CGで模型を作って遊びたいんだ！」

とはいうものの，パソコンでまっとうにCGをしようとすると，必ず「限界，制約，不便」というヤツがつきまとってきます。そして彼等は「パソコンCGで模型製作などムリだ！」と主張してはばかりません。

そこで私は考えました。

確かに正攻法で考えれば，プラモデル並みのデータをパソコンCGで描かせるのは不可能です。だからこの際，不可能となってしまう原因を見極め，そのうえでパソコンCGの新たな「方法」を作り出すべきであるという結論に達したのです。

かくしてANGELの開発はスタートしました。なかなか苦難の道のりでしたが，とりあえずシステムは現状でなんとか皆さんに発表できるようなものになったと思います（と勝手に解釈してよいのだろうか？）。

それでも，ほかのCGシステムにはない数々の特徴を持っているのが唯一の自慢です。

ANGELの紹介

「ANGEL」とは，ANimation GEneration Languageの略称です。直訳すればこのシステムは「アニメーション生成言語」となりますが，実際にはそれだけにはとどまらない多彩な機能を備えたものとなっています。

以下にANGELの持つ特徴を挙げてみます。

・BASICライクなコマンドやプログラムの入力実行環境

・数千面以上のポリゴンデータを一度に高速に描画できる驚異の新レンダリングアルゴリズム「Sバッファ法」を搭載！

・「変数」による演算機能を利用して即席にアニメーション作成プログラムを作ることができる。

・「形状データ」の対話的な配置，移動にとどまらず，形状データ同士の合成とファイルへの出力ができる

このように，ANGELは単にレンダリングのためのツールであるにとどまらず，モデリングシステムの一部機能やレンダリング環境の設定，物体・カメラ・光源の移動過程を指定またはプログラム記述することにより，インスタントにアニメーション作成する機能などを凝縮したシステムになっています（図1を参照のこと）。

しかも操作環境に関しては，BASICを強く意識したものになっています。すべての機能は対話的にコマンド入力することにより実行できますが，行番号をつけてプログラム化し，run コマンドで連続実行することもできます。ついでに変数の使用や式の記述もできますので，まさにBASIC感覚でアニメーション生成プログラムを作って動画像を連続作成することが可能です。

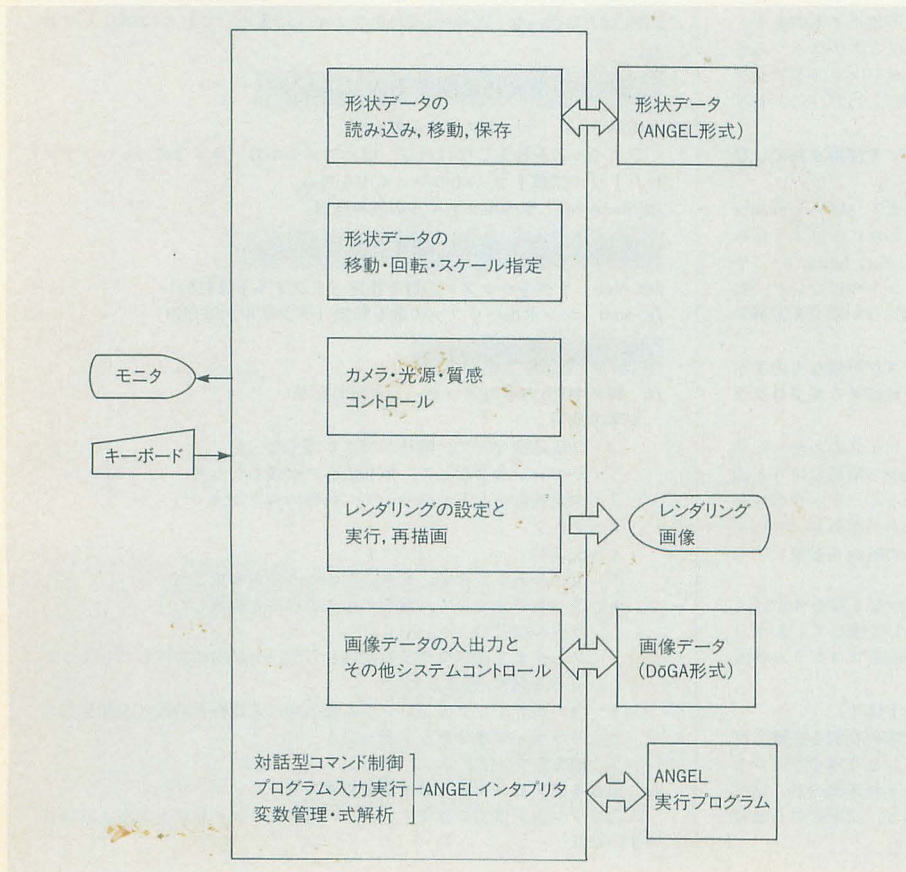
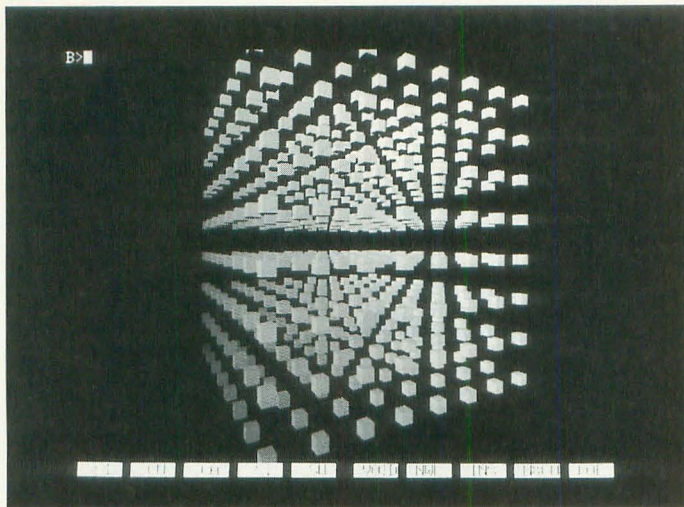
また，これまでのCGシステムのようにすべてのデータやパラメータを設定しなければ実行しないなどというワガママな子ではありません。描画に必要なパラメータはすべてあらかじめセッティングしてあります。だから初心者の方でもカンタンな操作でスムーズにCG画像作成を楽しむことができます。

そしてもうひとつ。

ANGELは，先に述べたとおり「CGによる模型製作」を強く意識して設計されたものです。よって操作実行環境がいくら優秀であっても，肝心の「レンダリング」において少しの形状データしか描画できないよ

図1 ANGELの機能構成図

図2

図3

うではお話になりません。さりとていくら大量の形状データが処理できても，最低限アンチエリアシング処理によるギザギザのない画像が生成できなければCGで絵を作る意味が半減してしまいます。

この一見ワガママな要求をなんとか満たすべく努力した結果，なんと「レンダリングアルゴリズムの独自開発」という方法で解決することができました。

その結果ANGELでは，アンチエリアシング処理を行ったうえで，6,144面という大量の形状データ(図2)を標準メモリのX68000で描画することが可能になったのです。

サンプルプログラムを実行してみよう！

それではANGELの使い方について簡単に説明していきましょう。

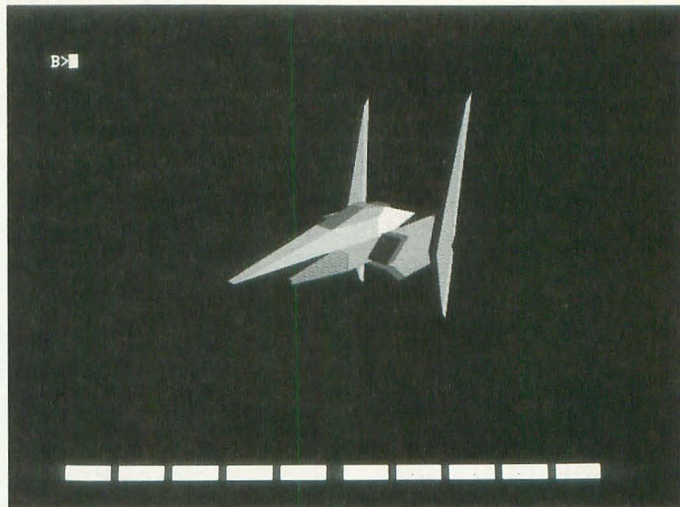
まずは起動についてですが，ANGEL実行のためには，angel.xの入った起動ディスクをAドライブにセットします。Bドライブには，フォーマットした直後のブランクディスクを入れてください。

この状態で，

A>angel

と入力すればANGELが立ち上がります。

ここで，もし立ち上がらない(「主記憶が不足しています！」が出る)場合には，CONFIG.SYSでカナ漢字変換ドライバ(ASK68K)を設定していることが考えられます。そこで，ANGEL実行ディスクでははずしておくようにすれば大丈夫です(もしくはメモリを増設するのが妥当かと思います)。

さて，ANGELが立ち上がったら早速なにかしてみたいと思うのが人情です(?)。とはいっても最初から難しい操作は避けて，シンプルにサンプルプログラムのロードと実行から始めましょう。

ANGEL実行用のサンプルプログラムには，fighter.angとcubic.angが用意されていますが，ここではfighter.angを実行してみることにしましょう。

それでは，ANGEL入力待ち状態(プロンプト“angel:”表示中)において，

load “fighter.ang”

と入力し，リターンキーを押してください(コマンドは必ず小文字で)。

すると，「プログラム読み込み開始：」のメッセージが出てただちに読み込みが終了します。ここでついついクセがでてlistと打ち込む人もいるでしょうが，大丈夫。ちゃんとリストが表示されます。

ロードが終われば次はRUNです。ここで，

run

と打ち込めば，いきなりレンダリングが開始されて図3の戦闘機(らしきもの)が表示されていきます。

が，なにか変だぞ！　よく見ればこいつ陰面消去してないじゃないか！　と一瞬思われてしまうようですが，ご安心を。

ある程度ペイントが終われば，ちゃんとあとからレンダリングして陰面消去アンチエリアシング付きの画像になっていきます(これはANGEL内蔵レンダラーのクセなのです)。

このようにして，ANGELのサンプルプログラム実行は終了，ちゃんと画像が生成できたわけですが，これだと普通のCGシステムのサンプルデモとそんなに変わりありません(……でもないか)。そこで，ちょっと面白いことをお教えしましょう。

実行が終わったあとで，

move(400, 0, 0)

と入力し，再度RUNさせてみてください。さてどのようになるかは見てのお楽しみです。

ANGELの描画実行方法

ここで，ANGELのサンプルプログラムを見てみましょう(表1)。このプログラムは，そのままANGELでの作画方法をトレ

ANGELの使い方

まず，オマケディスクから，

LH-E ANGEL

のようにしてANGEL.LZHを解凍します。

中にはANGELシステムとドキュメント，さらにサンプルデータ2種類分のファイル群が入っています。

まず，Bドライブにフォーマットした直後のディスクを用意し，

ANGEL B:

のように2HDディスクをバッファとして起動してください。手始めに，

load “fighter. ang”

run

と入力し動作を確認してください。

もちろん，2HDフロッピーディスク以外にもRAMディスク，ハードディスクに割り当てることができます。ハードディスクで使用している場合，仮にワークに使用するドライブがC：だったとすると，

A>ANGEL C:

という操作でハードディスクをバッファとして使用することができます(これをRAMディスクにすればさらに高速)。

ただし，長期間メンテナンスされずにあるハードディスクではディスクの空きエリアが細切れに散在している場合があります。こういう場合はかえってシーク時間が増えるためまっさらなフロッピーディスクよりも遅い場合があります。

いずれの場合もドライブのBUFFERという名前のディレクトリを作成しますので注意してください。

ースしています。

これらより，ANGELでは以下の手順でレンダリングを実行していることがわかります。

1) カメラ位置，光源位置などの設定
(システムリザーブ変数への値の代入)
c.pos＝(−500, 500, 1000)
c.tar＝(500, 0, 0)
l.pos＝(7000, 10000, 10000)
l.tar ＝(500, 0, 0)
2) 描画モードの設定
drawmode QUICK
3) 物体形状データの読み込み
m.rd "f_body01.mdl"
4) 物体の色(質感)の設定
(システムリザーブ変数への値の代入)
a.rgb＝(0.7, 0.7, 0.7)
5) 描画の実行
draw

1)以下は3)から5)までの繰り返しとなっていますが，普通のCGシステムなら一度形状データを読み直せば前の画像との陰面消去はできないはずなのに，ANGELでは平気で処理していきます(ゔーん，なんと非常識な)。

このようにANGELでは，描画に必要なパラメータをすべて変数化することにより余計な設定コマンドを減らすとともに，計算式中へのダイレクトな参照が行えるように配慮されています(表1参照)。

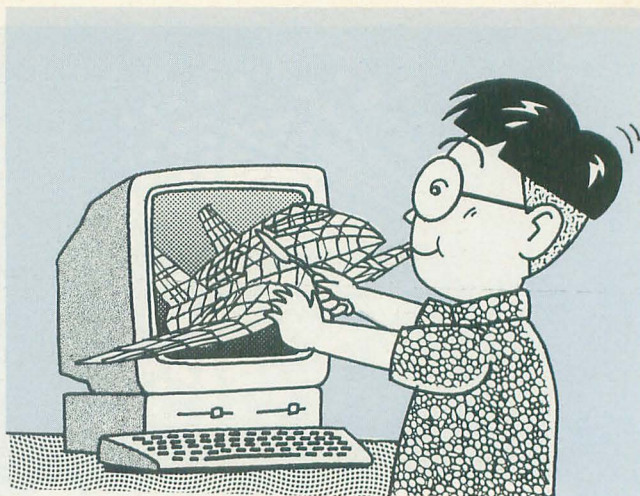
したがって上記の例で描画に使っているコマンドは，形状データファイル読み込みのためのm.rd (mdlread) コマンドと描画モード設定のためのdrawmodeコマンド，そして描画実行のための drawコマンドの3つだけです。

そうです!「Simple is best!」こそがANGELの身上です。要は面倒くさいことがキライなんだよーっ，てかっ。

変数について

上の「システムリザーブ変数」のように，ANGELでは独自に変数を扱い，式の記述・計算・代入といった操作が実行可能になっています。ということで，ANGELでは表3のような変数を使ったプログラムも実行できます。

ここで，ANGELで利用できる2種類の「変数型」について説明いたします。

そのひとつは「var型」と呼ばれるもので，これはごく普通の浮動小数点変数です。もうひとつは，x・y・zの3つの値を同時に扱う「vtx型」というものです。

これらの変数の宣言は，以下のように行います。

var var1, var2, var3
vtx vtxtype

また，これらの変数の代入は，

vtxtype＝(var1, var2, var3)

や以下のようになります。

var1＝vtxtype : x
var2＝vtxtype : y
var3＝vtxtype : z

このように，vtx 型変数への代入は3つの数値やvar型変数(式の記述も可)を上の書式で記述します。

逆にvtx型変数をvar型変数に代入する場合は，: x，: y，: zといった修飾子をつけて記述します。

表1 サンプルプログラム fighter.ang

```
100 c.pos = ( -500, 500, 1000 )
110 l.pos = ( 7000, 10000, 10000 )
120 c.tar = ( 500, 0, 0 )
130 l.tar = ( 500, 0, 0 )
140 param
150 drawmode QUICK
1000 m.rd "f_body01.mdl"
1010 a.rgb = ( 0.7, 0.7, 0.7 )
1020 draw
1030 m.rd "f_canop.mdl"
1040 a.rgb =( 0.0, 0.2, 0.7 )
1050 draw
1060 m.rd "f_tank01.mdl"
1080 a.rgb = ( 0.0, 0.7, 0.5 )
1090 draw
1100 m.rd "f_tank02.mdl"
1110 a.rgb = ( 0.55, 0.55, 0.55 )
1120 draw
1130 m.rd "f_nozl01.mdl"
1140 a.rgb = ( 0.8, 0.2, 0.2 )
1150 draw
1160 m.rd "f_nozl02.mdl"
1170 a.rgb = ( 0.55, 0.55, 0.55 )
1180 draw
1190 m.rd "f_fin01.mdl"
1200 a.rgb = ( 0.6, 0.6, 0.6 )
1210 draw
1220 wipe
1230 redraw
```

表2 システムリザーブ変数一覧

カメラ設定	c.pos	視点 (vtx型)
	c.tar	目標点 (vtx型)
	c.zoom	ズーム角(var型)
光源設定	l.pos	光源点 (vtx型)
	l.tar	目標点 (vtx型)
	l.rgb	光源色 (vtx型)
	l.gain	光の強さ(var型)
	l.kind	光の種類(var型)
質感設定	a.rgb	色 (vtx型)
	a.lrg	発光色 (vtx型)
	a.dif	減衰率 (var型)
	a.pspc	ハイライト係数(var型)
	a.pspx	ハイライト乗数(var型)
	a.amb	環境光強度 (var型)
	a.opa	透過率 (var型)
	a.gain	色の強さ (var型)
	a.lga	発光強度 (var型)

表3 変数による計算実行サンプルプログラム

```
10 var  i , j
20 i = 5
30 j = 10
40 i = ( i + j * 10 ) * i
50 print i
```

ANGELのコマンド解説

ここで，ANGELのコマンドについて説明いたします。なお，文中の〈ファイル名〉，〈変数名〉，〈式〉については表4を参照してください。

1) プログラム読み込み・書き込み・編集・実行系コマンド

load "〈ファイル名〉.ang"
save "〈ファイル名〉.ang"

これはANGELの描画実行プログラムを読み込みまたは保存するためのコマンドです。ここで，〈ファイル名〉の後ろの拡張子には".ang"をつけてください。

list , llist , rnu
delete 〈行番号〉

リスト表示(印字)，行削除，プログラム実行のためのコマンド群です。

list，llistおよびrunコマンドではパラメータの設定はできませんが，deleteコマンドでは削除する行番号を指定する必要があります。

2) 変数定義・変数値表示およびプログラム制御

var 〈変数名〉，〈変数名〉，……
vtx 〈変数名〉，〈変数名〉，……

これは先ほど紹介した変数定義のためのコマンドです。var型およびvtx型についてはすでに説明したとおりです。

print 〈式〉，〈式〉，……

式の値を表示するためのコマンドです。

loop i=a, b
endloop

いささか強引な例ではありますが，ここではi，a，bはともにvar型変数です。しかしa，bについては〈式〉とすることができます。

いずれにしてもお馴染みのループ命令です（ただしa＜bで，＋1増分に固定されているので注意のこと！）。

3) グラフィック表示およびレンダリング関係

drawmode 〈QUICK または WIRE〉

描画モード（レンダリングかワイヤー表示）の設定を行います。

draw, store, redraw, zinit

描画の実行を行います。draw では最終レンダリング処理付き，storeではSバッファへのデータの蓄積のみです。

redrawでは，描画時に蓄積された陰面消去情報をもとに，高速にレンダリング画像を再描画していきます

zinitは，過去にレンダリングして蓄積された陰面消去情報（Sバッファの内容）を消去し，新しいシーンの描画を行うためのコマンドです(重要！)。

gload 〈式〉，"〈ファイル名〉"
gsave 〈式〉，"〈ファイル名〉"

画像データのロード/セーブを行います。〈式〉は画像No. を表します。画像データはDōGA・CGA画像フォーマットで圧縮保存されます。

wipe, logo

画像消去およびシンボル文字表示コマンド。logoのパラメータはX-BASICのSYMBOLと同じです。

4) 形状データの読み込み・保存およびアフィン変換・変換行列操作

mdlread "〈ファイル名〉.mdl"
mdlwrite "〈ファイル名〉.mdl"

ANGEL形式の形状データを読み込み・保存します（省略名m.rd, m.wr)。

なお，ANGEL形式の形状データは拡張子を".mdl"として区別します。

move　(〈式〉，〈式〉，〈式〉)
scale　(〈式〉，〈式〉，〈式〉)
center　(〈式〉，〈式〉，〈式〉)
rotx　〈式〉
roty　〈式〉
rotz　〈式〉

これらは読み込まれた形状データに移動・回転・スケーリングをかけるものです。centerは回転の際の中心位置を設定するのに使います。

matpush , matpop , matinit
affine

これらはすべてパラメータなしです。matpush，matpopは上記の移動命令を一時スタックに待避・呼び出しするものです。matinitはそれまでの移動命令を取り消し，物体を元の位置に戻します。affineでは逆に現在位置に物体を固定します。

5) その他の命令

system "〈Human68k コマンド〉"

Human68kのコマンドを実行します。メモリが2Mバイトに増設されているなら，ed.xやpic.rの実行が可能です。

files "〈dir パラメータ〉"
chdir "〈cd パラメータ〉"

各ドライブのファイル名を調べたいときやディレクトリ変更に使います。パラメータは各DOSコマンドと同じです。

contrast 〈式〉

画面のコントラストを調整します。

cls , pause , param

テキスト画面消去，写真撮影用ポーズ画面，システムパラメータ表示コマンドです(パラメータ不要)。

quit

ANGELを終了させます。

(1)ANGELの形状データフォーマット

以上に挙げたコマンドを使えばANGELを操作できるわけですが，実際にANGELを使っていくうえで必要になると思われるテクニカルなアドバイスをしておくことにしましょう。

まず，ANGELで描画に使われる「形状データ」の形式なのですが，残念ながら現在のところオリジナルデータフォーマット(".mdl"形式)のみとなっています。

実はDōGA・CGA規格に準拠できるようなデータコンバータとセットにするつもりでしたが，開発がとうとう間に合いませんでした。

ということで，ANGEL形状データフォーマットに関する説明を図4に示します。見てわかるとおり「格子状データ構造」という非常にユニークな形態を取っていますが，これは大量の面を少ないデータで表現できることから採用したものです。

(2)ANGELでの物体移動指定

ANGELにおける物体の移動指定は，move，rotx，roty，rotzといったコマンドにて行います。ところが，これらの移動は基本的にどんどん蓄積されていきます。

すなわち，1回目に「X方向に200」とし，2回目に「Y方向に400」と指定すると物体はそのとおりに動きます（専門用語でいうところの「相対移動方式」なのです)。

これを取り消して物体を元に戻したいときは，matinitというコマンドを用います。また，疑似的に階層構造を実現するためのmatpush，matpopというコマンドもありま

表4 ANGEL識別子能力表

〈ファイル名〉	12文字までの英数字の集まり。ただし画像ファイルの場合，ナンバリングを考慮して6文字くらいにしておくとよい。
〈変数名〉	先頭文字のみ英字，あとは英数字およびピリオドが使える。文字数に制限はなく，全文字判定の対象となる。英文字大小の区別あり。
〈式〉	数値，変数，定数のほか，これらと演算子およびカッコと組み合わせたもの（カッコのネストは20まで)。

す。

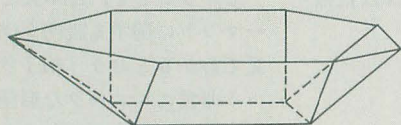
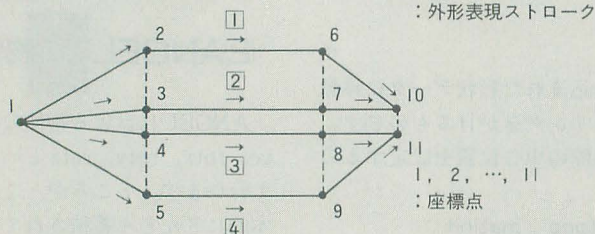
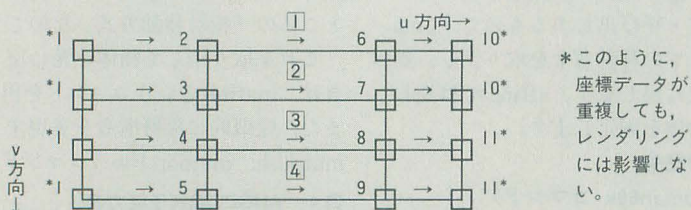
(3)レンダリングに関する注意

ANGELのレンダリングは，あとで述べる「Sバッファ法」を採用しているため，数々のユニークな機能があります。

まず，Sバッファが初期化されない限り，形状データをあとから追加描画しても問題なく陰面消去が行われ画面に表示されます。

その代わり，新しくシーンを描画する際には必ずSバッファをzinitコマンドにより初期化しなければなりません。

また，一度画面が消去されてもSバッファの内容が残っている限りredrawコマンドでレンダリング画像を復元することができます。この機能を利用して，バックに別な絵を表示させることにより画像α合成が可能となります。

あと，Sバッファを確保するための領域を任意のドライブに設定する機能もあります。それはANGEL起動コマンドの第1パラメータにドライブ名を指定すればOKです（RAMディスクでの実行は速い！）。

驚異のレンダリング性能！「Sバッファ法」のタネ明かし

さて，ANGELではなぜ少ないメモリで大量の形状データを高品質にレンダリングすることが可能になったのでしょうか？

それは，現在研究中の新型レンダリングアルゴリズムである「走査線ストロークバッファ(Sバッファ)法」をシステムに採用した結果導き出された性能であるといえます。

このアルゴリズムは，Zバッファ法を発展させたものです。Zバッファ法はスクリーン上の全ピクセル（画素）分に対応するバッファ領域さえ確保できれば，「無限」に形状データを処理できる方法です。ところがアンチエリアシング処理を考えると，バッファ領域に恐ろしい量のメモリが必要となることから，パソコンでは敬遠されてきました。

しかし，このZバッファに記憶されるデータを，画像データと同じように「圧縮」できる方法があるとしたらどうでしょうか？私はその可能性があることに気づき，研究の結果以下の方針でバッファを組めば有効であることをつきとめました。

①そのバッファは，走査線ごとに独立した領域に展開される
②そのバッファを構成する1レコードは，あるポリゴンと2本の走査線の交点による区間のX値およびZ値で構成される
③これらのレコードは走査線上で互いに重なり合うこと，交叉することを許される
④上記の結果画面上に見えないレコードができたら，それは排除することができるものとする

この方針により構築されたものが「走査線ストロークバッファ法」なのです。その具体的な内容について表5に示します。

図4　格子状形状データフォーマットの構造

(1)Sバッファ法の処理構造

それではSバッファ法がどのようにして処理を行うか説明していきます。

Sバッファ法の処理の流れは明快です。

処理は図5のように大きく2つに分かれます。

まず個々の形状データに対し，各走査線と交点を求め，それを当該Sバッファのレコードとしてどんどん追加していきます。

次に，登録されたレコードの内容を順次スキャンラインZバッファに描画し，陰面消去とアンチエリアシング処理を行います。

この2つの処理により，1枚のレンダリング画像が完成されるわけです。

(2)Sバッファ法の特徴について

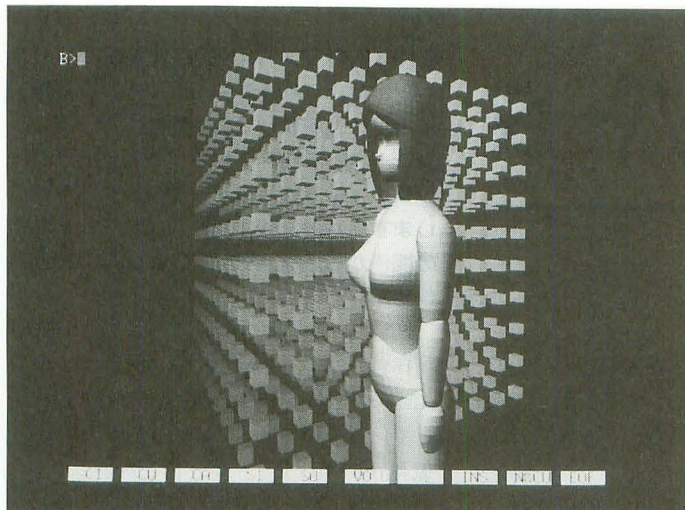
このSバッファ法をANGELの中にインプリメントしていろいろ実験を行った結果，以下の特徴があることがわかりました。

1)　少ないメモリで大量の形状データを処

表5　Sバッファ構成データ(ストローク)の内訳

xsp1，xep1 xsp2，xep2	ストローク開始点，終了点の位置
zstart，zend	ストローク開始点，終了点のZ値
red green blue	ストロークの色（RGB値）
transparency	ストロークの透明度

図6

図7

理することができる

2) 大量データ描画時に対しては，比較的高速に描画が実行できる

3) バッファにデータが保存されている場合，画面が消去されてもその内容を高速に再描画することができる

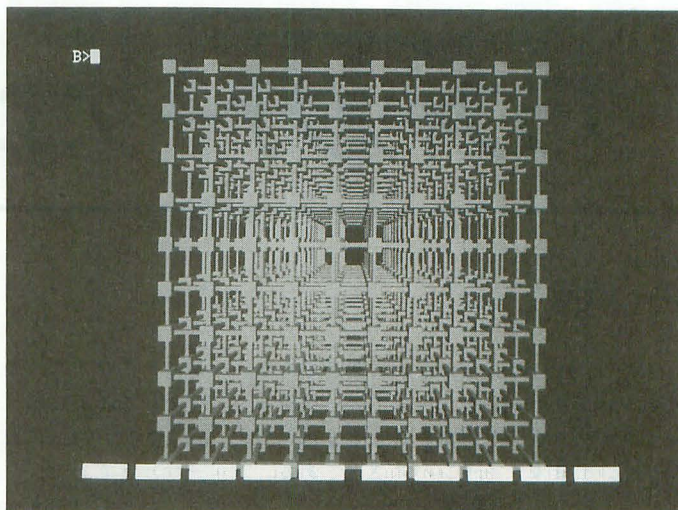
4) ハードディスクなどの大容量外部記憶装置を効率よくバッファに使用することができる

これらは，図6，7にあるサンプル画像を見れば納得していただけると思います。とりわけ，4)の特徴によりこの方式では1万面を越える形状データの描画を実行することが可能になりました。図6の人体データは13,000面ほどの数ですが，ハードディスク使用で1時間半で描画します。しかも使用したバッファ量は1.4Mバイトくらいで済んでいます。しかも，これらの実測値はいずれも数値演算プロセッサなし，外付けの40msecのHDを利用した結果のものですから，まだまだ高速化できる余地は残されているのです。

おわりに

なんだかんだガチャガチャ書いてしまいましたが，実はこれでもまだ書けなかった部分がたくさんあります（たとえばアニメーション生成プログラムの作り方とか）。

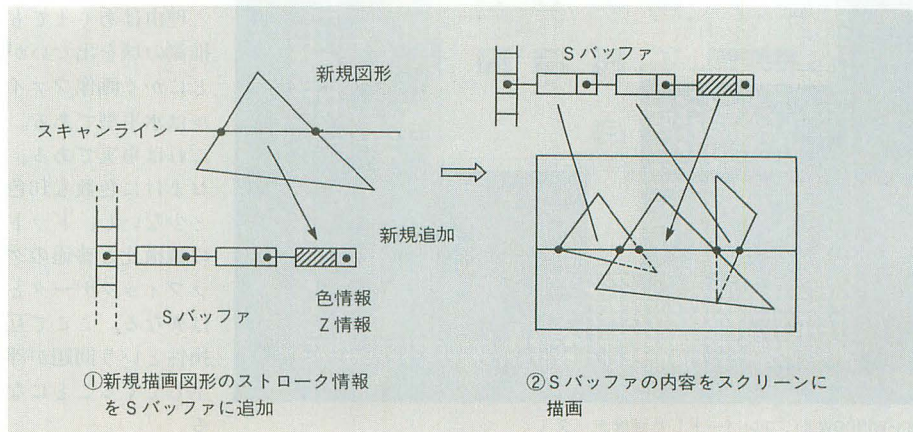
しかし，まだ現時点のANGELでは完成にはほど遠い状態であることは確かです。ループ文には制約がありすぎるわ，条件分岐はないわで，まだ言語インタプリタとしても手を入れる余地は山ほどあります。

さらにCGシステムとしても，「階層構造」をサポートしていないという問題もあります。しかし，階層構造による運動表現には以下の問題が存在します。

たとえば階層構造で歩行動作を作ると，「足が地面を蹴って前に進まず，ただ空中でバタバタするだけ！」になってしまいます。これは階層構造が実際の重心移動をともなった運動をトレースできないために起こるものです。最近，プロの世界でもようやくこのことに気づき始めたみたいです。

ということで，次のバージョンでは，CGの運動記述における新しい方法の提案をひっさげて登場するつもりです。もっとシンプルに，もっと自然な運動を定義・実行する方法はあるのです！

あとはレンダリングで点光源，自動スムーズシェーディングとマッピングを，モデリングではDōGAのものが利用できるようにするほか，オリジナルなモデリングツールぐらいサポートしようかと考えています。

それにしてもまだまだ説明が足りないというのに誌面が尽きてしまいました。この埋め合せは必ずしますのでご容赦を。

最後に，ここまで読んでいただいた皆様に感謝するとともに，発表とANGELの配布の機会を与えていただいたOh!X編集部の皆さんに感謝します。

そして「幻想工房ぷろせにあむ」唯一のメンバーで私のよき理解者である尾崎君に多謝！ いろいろ協力してくれる大阪工業大学グラフィック研究会の後輩たち，東京CG連の太田さん，そして7枚ポーカーを教えてくれたDōGAプロジェクトの皆さんに感謝します（いつもお世話になってます）。

では皆さん，ANGELシステムを使ってみてください。

〈参考文献〉

1) 森山：「リアルなCGモデルを自由に表現するための新しいレンダリングアルゴリズムの提案」，PIXEL1990年6月号

2) D. Rogers（山口訳）：「実践コンピュータグラフィックス，基礎手続きと応用」，日刊工業新聞社，(1987)

表6 描画例とその面データ量および描画所要時間

描画例	面数	所要時間
図2(色立体)＊	6,144	1時間4分
図3(戦闘機)	180	12分
図6(人体モデルのみ)	13,000	1時間34分
図7(ジャングルジム)	22,800	3時間12分

＊このモデルに関してのみRAMディスクによる実測値（FDでは2時間14分かかる）
他はハードディスク（40msec）による実測値

図5 Sバッファによるレンダリングの手順

Graphic

SX-WINDOW&キャンバス.X用

グラフィックデータコンバータ

Oh!X初のSX-WINDOW周辺アプリケーション。これまでのグラフィック処理アルゴリズムを使って65536色のグラフィックデータからデスクトップを彩るグラフィックウィンドウ用のデータファイルを作成します。なお，残念ながら画面モードなどの関係からSX-WINDOW上では動作しません。

Tan Akihiko
丹　明彦

SX-WINDOWはなかなかに曲者である。グラフィックデータをテキストデータやバイナリデータと同じ感覚で扱えるとは，天晴天晴。つまり“キャンバス.X”である。

キャンバス.Xの秘密？

キャンバス.Xのアイコンをダブルクリックすると，“Canvas”と書かれたウィンドウがぽんと開く。そして見たい画像のファイル（つまりアイコン）を運んでいってキャンバスウィンドウに放り込む。するとウィンドウの中に絵が出現する。

表示する絵のサイズに合わせてウィンドウも大きくなったり小さくなったりする。当然，ウィンドウは画面のどこにでも移動できる（中の絵もいっしょに動く，という意味)。何枚でもウィンドウは開けられるし，重ね合わせも完璧である。当たり前のことだが，うーん賢い賢い。

と，持ち上げておいてからいうのもなんだが，ほめっぱなしというわけにはいかない部分もある。まず，16色の絵しか扱えない。これはSX-WINDOWが768×512ドットモードで動く以上，しかたのないことである。ウィンドウシステムを512×512ドットモードで設計するというわけにもいかなかったのだろう（やればなんとかなったはずだが)。まあこれはいい。しかし困ったのは，キャンバスに放り込むファイルのフォーマットである。

これまではimg_saveやimg_loadを使って取り扱う形式が事実上X68000グラフィックの標準フォーマットであった（単にG-RAMの内容をベタで格納しているともいう)。ところがどうしたことか，X68000のG-RAMが垂直型であるのに対し，キャンバスが読み込む画像ファイルは水平型なのである。この差はいったいどこから出てきたのだろう。

SX-WINDOWはテキストVRAMをベースにして動作しているのである。だからグラフィックデータもそれに合わせた設計になった，ということだろうか。

というのも，ウィンドウシステムでもっとも厄介な部分のひとつは，複数のウィンドウの重ね合わせであり，クリッピングであるからだ。テキストと画像データのデータ構造が著しく違っていると，こうした処理が繁雑になるばかりである。ならばいっそのこと，初めから画像データも水平型で持っておけば，ほかの（テキスト表示をしている）ウィンドウとの重ね合わせをするのに，特別な判定および表示ルーチンを作る必要がなくなるからである。これなら少し楽ができる。

SX-WINDOW上にコンバートした画像データ

理由はあくまでも推測の域を出ないが，とにかく画像ファイルは水平型である。これは事実である。おまけに色数も16色と少ないし，ドットの縦横比も普通のグラフィックデータとは異なる。ここで互換性という問題が浮上してくることになる。

16色でプレーン別（つまり水平型）G-RAM構成というのは，PC-9801などが採用しているものと同じである。SX-WINDOWは768×512ドットモードで動作していて，このモードだとピクセルはほぼ正方形になる。PC-9801は640×400ドットで，やはりピクセルはほぼ正方形である。

ところでX68000でよく使われるグラフィックモードは256×256または512×512ドットモードで，ピクセルは横長である。X68000上で作成されたグラフィックデータは，ゲーム画面からレイトレーシングに至るまで，9割はこの横長ピクセルでできているはずである。

したがって，Z'sSTAFFなどで描いた絵をそのままSX-WINDOWに持ち込もうとすると，絵がつぶれて縦長になってしまう。これはうまくない。X68000ユーザーには，768×512ドットモードはもちろん，16色で絵を描く人などほとんどいない（と思う)。

およそX68000的とはいえないこのタイプの画像データを大量にストックしているX68000ユーザーがどのくらいいるかなどという意地悪な疑問はさておいて，このままではせっかくのキャンバスも宝の持ち腐れである。使い回しのきかない画像データは，いくらあっても役に立たない。とはいえ，今後はこれがSX-WINDOW上でのX68000の共通フォーマットのひとつになる。

キャンバスはきっとおいしく使えるはずである。キャンバスを働かせるだけでビジュアルシェルのときに比べて画面がぐっと華やぐ。映画でよく見るアメリカのビジネスマンみたいに，家族の写真をオフィスの机上（デスクトップ）に置いて仕事ができるのである。日本のパソコン少年なら，さしずめ女の子のイラストといったところであろうか。そういう使い方にどのくらいの人が憧れているかという野暮な質問は無視しよう。

で，これまでに膨大な量の蓄積があるであろう画像──65536色，512×512ドットモード。あのPIC.Rだってこのモードが基本である──をなんとかして利用できるよう

にしたいわけである。
問題点は3つ。
・65536色をどうやって16色に落とすか。その際，元絵のイメージを損なうことは極力避けたい（色の印象が極端に変わってはいけない）。
・ドットの縦横比の問題。元絵のドットは横長，対してキャンバスはほぼ正方形。これも見た目にはわからないようにうまく修正したい。
・元絵はいうまでもなく垂直型のデータである。これを水平型のデータにしなくてはならない。

データコンバートの方法

さっそく解決編といこう。
3番目の問題（垂直型→水平型の変換）は，さんざん大騒ぎしたわりには，解決法は取るに足らない。簡単なビット操作ですむ。で，残る2つであるが，

バックナンバーは大切に

なんのことかとお思いのあなた，解決法はすでに昔のOh！Xにあったのだ。1番目の問題（65536色→16色）は1988年11月号のプリンタ特集で桒野氏によって紹介された65536色の画像を自然に2階調に落とすアルゴリズムが，そして2番目の問題(768ドット→512ドット）は先月号のBASIC特集で僕が書いたグラフィック拡大縮小アルゴリズム(う〜むなんとも手前ミソ）が，たちどころに解決してくれるではないか。バックナンバーは大切に。
詳しい解説はそちらに譲って（おっとひどい手抜き），さっさと今回のプログラムの使い方を説明しよう。付録のフロッピーディスクに収録されているsxconv.xがそれだ。以下はコマンドモードで動かすことを前提にする（SX-WINDOWから呼び出してもうまくいかない）。
1） まず画面モードを65536色・512×512ドットモードにする。
　SCREEN 1,3,1
2） 手持ちの画像ファイルを表示する。たとえばPIC.Rなら，
　PIC xxxx.PIC
3） SXCONV.xを起動する。パラメータとして与えるファイル名には，拡張子をつける必要はない(たぶんエラーになる)。
　SXCONV xxxx
4） マウスカーソルが出てくるので，左上と右下の座標をポイントして左クリックする。左上の座標をポイントしたあとでも，気に入らない場合は右クリックで初めからやり直すことができる。
5） しばらく待つとコマンドモードに戻ってくる。そこでディレクトリを取り，
　xxxx.PIX
という名前のファイルができているのを確認する。
6） SX-WINDOWに入り，キャンバス.Xをダブルクリックする。出てきたキャンバスウィンドウに，いま作ったxxxx.PIXのアイコンを放り込む。きっと望みどおりの画像が出現することだろう。

sxconv.xのアルゴリズム

本プログラムsxconv.xは，65536色・512×512ドットのグラフィック画面から任意の矩形領域を切り取って，16色・768×512ドットに合わせた画像に変換し，キャンバスのフォーマットでファイルにセーブするプログラムだ。リスト1がCのソースリストである。
具体的には，まずグラフィック画面のマウスで指定した領域を切り取り，拡大アルゴリズムを使ってx方向のドット数を1.5倍する（512：768＝1：1.5)。次に階調落としアルゴリズムを使って65536色を16色に落とす。桒野氏のアルゴリズムでは8色にすることは簡単だが（3プレーン分実行すればいい)，残りの1ビットの処理に困ってしまう。今回はRGBのうちもっとも輝度の高いと思われるG(グリーン)の部分を2ビットで処理することで16色に対応した。画像の色の傾向から最適な16色を選ぶ手もあるが，このままでもまあまあの画質はあるので手抜きさせてもらった。それをプレーンごとに分解してファイルに記憶する。ファイルには画像の情報を記録するヘッダ部をつける。なお残念ながら(？)，sxconv.xで作った画像を表示するには，SX-WINDOWとキャンバス.Xが必要である。
ところで，理由は不明だが，キャンバス.Xは横最大640ドット，縦最大480ドットまでの画像ファイルしか受け付けない（最小は横・縦それぞれ128・64ドット)。ちなみにIBM PCの解像度もこれである。あまり大きな画像は入りきらないわけだ。横640ドットというのがいかにもPC-9801ぽい？
というわけで，エラーが起きないようにsx

色表示の基礎知識

基礎知識として，コンピュータでは「色」をどのように扱っているかをお話しておくことにしよう。
まず色は有限の整数値で表す。たとえば65536色モードなら，色コードは0〜65535の値をとる。有限の整数値なのだから有限のビット数で格納することもできる。8色なら1ピクセル当たり3ビット，4096色なら12ビット，65536色なら16ビットというぐあいである。
1枚の画面には数万から数十万のピクセルがある（たとえば512×512ドットモードでは，実に26万余りのピクセルを用意しなくてはならない）が，上のようにあるていど簡素化した色の表現を用いることで，グラフィックメモリに掛かるコストを抑えることができる。近年は，RAMもハードディスクに代表される大容量外部記憶装置も安くなったので，必然的に多色化傾向が進んでいるというわけである。ほんの数年前までは，白黒2色表示でも十分に感動していたことからすると，感慨深いものがある。
ついでに「パレット」についてもお話しておこう。色の話からもわかるように，グラフィックメモリが膨れ上がるのを抑えるためには，1ピクセル当たりのビット数を減らすのが効果的である。たとえば1ピクセル4ビット＝16色モードを採用したコンピュータに話を限ってみよう。このコンピュータが同時に画面に出せる色は確かに16色である。しかし，パレットを使えば，もっとたくさんの（という表現が不適当なら「微妙な」でもよい）色を出せる可能性が出てくる。いい換えれば，1ピクセルの4ビットに4ビット以上の表現力を持たせることができるのだ。
多くの画像は，使えるだけの色を使ってなどいない。極端な話，65536色モードで65536色をフルに使い切った絵というものは事実上存在しない。本当によく使う色はごく限られた何色かである。というわけで，手持ちの16色によく使う色を割り当てようという発想が生じる。なにも手を加えなければ原色にすぎなかった色が，パレットのおかげで中間色に変身する。たったこれだけのことでも，絵の表現力は相当に上がるものである。
そして，パレットに使うメモリ容量は高々数十バイト。ピクセル当たりのビット数をたったひとつ増やすだけでも数十Kバイト余計にグラフィックメモリを消費することを考えると，同じ多色化表現でも，パレットのほうが数段コストの安い方法である。
さらにグラフィックメモリの構造についても少々説明をしておこう。
X68000のG-RAMは垂直型の構成をとっている。つまり，「1ピクセル＝1ワード」の原則を貫いており，各ピクセルは完全に独立している。65536色モードでは，1ピクセル＝16ビットがメモリ空間に順番に並んでいる。
これに対し，水平型のG-RAM構成というのもある。ピクセルの各ビットをそれぞれ集めてプレーンを数枚作り，メモリ空間には別々に配置する。たとえば16色モードの場合，1ピクセルは4ビットだから，4枚のプレーンができる。たくさんあるピクセルの第0ビットばかりを集めてプレーン0に，第1ビットをプレーン1に，……という調子でメモリに格納する。1ピクセルを構成する4つのビットは，メモリ空間上の4地点に飛び散っている形になる。PC-9801やX1，X68000のテキストなどはこの水平型のG-RAM構造を持っている。

conv.xのほうに手ごころを加えてある。安心して使っていただきたい。

システムコールなどはMacintoshライクといわれるSX-WINDOWも，グラフィックまわりはまるでPC-9801みたいだ。

キャンバス.Xのフォーマット

簡単にキャンバスの画像ファイルのフォーマットを解説する。図をご覧いただきたい。x×yドットの大きさの画像を格納したい場合について説明する。ソースプログラム中の変数名とは違っているが，そこは例によって読み換えをお願いすることにしよう（このフォーマットは僕が勝手に調べたものなので，もしかすると真実とは食い違っているかもしれない。しかし，まあプログラムは正常に動いているようなので，きっとこれでいいのだろう)。

ファイルは48バイトのヘッダと，第1～第4プレーンのデータで構成されている。まずはヘッダからいってみよう。

最初の4バイトは“TX16”という文字列を表すASCIIコードが入っている。これはまあ縁起ものと思って入れておこう。もちろん画像ファイルをほかのファイルと識別するためのものである。

次の4バイトはファイルサイズから8を引いた値である。画像データは，ウィンドウの移動や重ね合わせで何回も参照されるので，いったんメインメモリに格納すると思われる。その領域を確保するために必要なデータサイズなのであろう。このあたりは，OSに組み込むアプリケーションを感じさせる。ここでファイルサイズ“－8”というのは，この時点ですでにファイルの先頭から8バイトを読み込んでいるからで，この値は残っている部分のサイズを表していることになる。

次の4バイトには0が入る。x，yのオフセットが2バイトずつ入っているのではないかと思われるが，確証はない。ウィンドウはあちらこちら動かすものだし，まあ0を入れておけば間違いないようである。

その次の4バイトは画像の大きさである。「真のx方向のドット数」と「y方向のドット数」がそれぞれ2バイトずつ入っている。“真の”がどうも引っ掛かるという方へ。実は垂直型から水平型に変換する段階で，「仮のx方向のドット数」とでもいうものができるのである。処理の都合上だろうが，x方向に16ドットを単位としたデータをメモリに格納する。だから，いったん画像の右端に数ドット分ダミーの画像を付け足して，それをファイルに記憶する。

このx方向のサイズが「仮の……」である。たとえば，横方向243ドットの画像は，13ドット分を右に付け足し，256ドットの画像のつもりでファイルにセーブする。真のドット数は243で，仮のドット数は256である。そうして表示の段階では，ダミー画像を切り落とした，真の画像をウィンドウ内に収めて表示するというカラクリになっている。

次の32バイトはパレット。16色分のパレットを，1色につき2バイトの色コードで記録する。ここまでがヘッダ。

49バイト目から先は各プレーンの内容を記録する。そのバイト数はもちろん画像の大きさによって異なる。sxconv.xの場合は，順に青プレーン・赤プレーン・緑プレーン1・緑プレーン2が記録される。4プレーンとも大きさは同じだ。

図1 キャンバス.Xのグラフィックデータのフォーマット

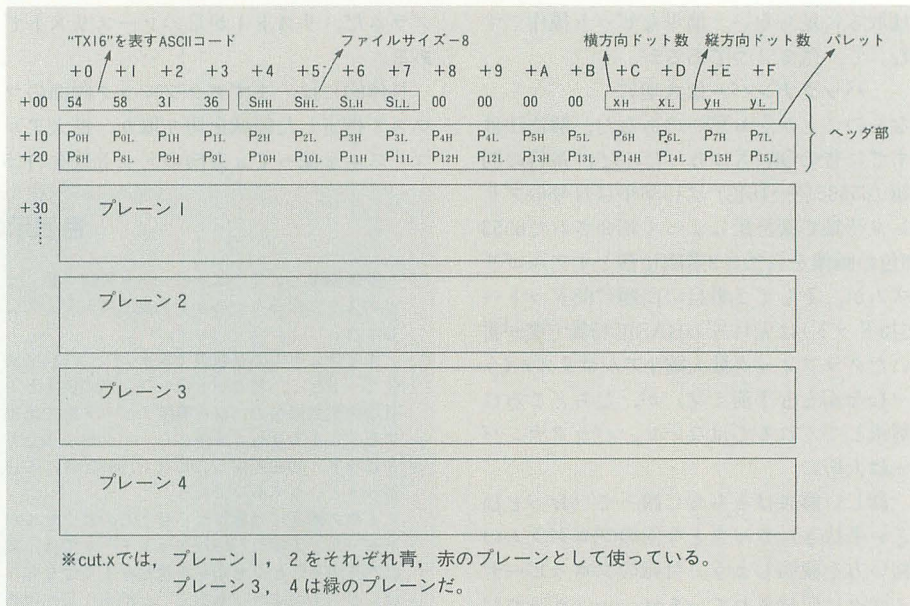



図2 拡大ルーチンを使う理由

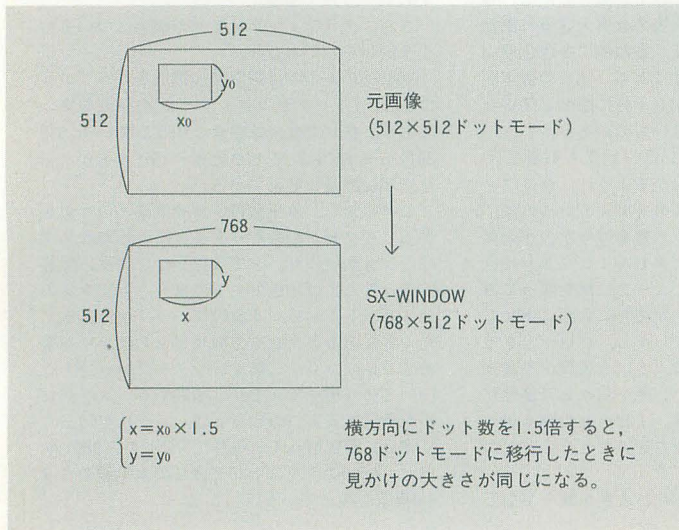
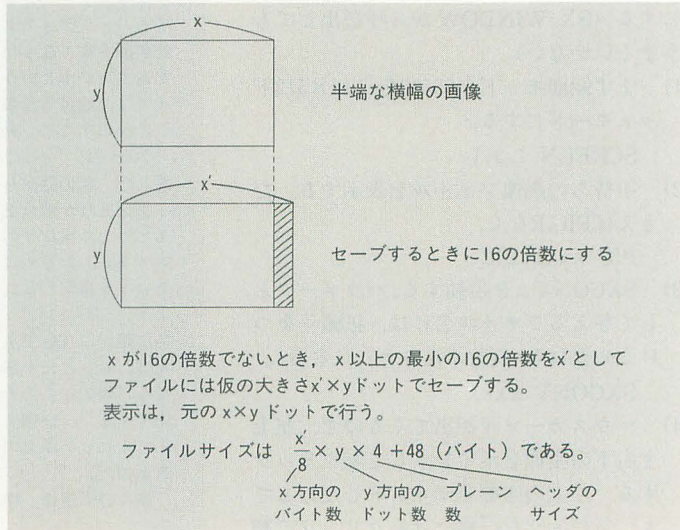



図3 半端な横幅を持つ画像の処理

＊

これで一応，いままで溜めたグラフィックデータを無駄にせずにすむ。しかし，コマンドモードで動かすというのがなんとも情けない。やはりSX-WINDOW上からすべてできるというのが理想であろうと思うのだが，いかがか？　いやいや，理想をいうなら，SX-WINDOWにもPICを！　といったところなんだろうな，きっと。それに，キャンバスの役目はグラフィックを表示するだけで，そこから先の活用はできない。表示ができること自体はたいしたものであるが，さらなる活用となると，65536色への対応は考えなくてはならないかもしれない。

リスト1　sxconv.xソースリスト

```
 1: /*********         SX-WINDOW 用ユーティリティプログラム sxconv.x         *********/
 2: /*                              (機能)                                        */
 3: /*              フルカラー画面(65536色)から任意の矩形領域をカットし           */
 4: /*          SX-WINDOW (の キャンバス.x) 仕様に変換してファイルに出力する      */
 5: /*    ドットの縦横比も修正する (65536色 512×512ドット→16色 768×512ドット)     */
 6: /*                              (使い方)                                      */
 7: /* 1) スクリーンモードを65536色モードにして(コマンドラインからは screen 1,3,1) */
 8: /* 2) 画像をロードしておく(PIC.Rなどで)                                        */
 9: /* 3)コマンドラインから起動する(ファイル名に拡張子は不要)                     */
10: /*      A> sxconv filename                                                    */
11: /* 4)マウスで矩形領域をポイント(2頂点を左クリック)すると、ファイル            */
12: /*      filename.PIX                                                          */
13: /*    が生成される                                                            */
14: /****************************************************************************/
15:
16: #include        <stdio.h>
17: #include        <graph.h>
18: #include        <iocslib.h>
19:
20: /****** キャンバス.x が読み込める画像のサイズには制限があるらしい ******/
21: /******              ( 128×64 ～ 640×480 ドット )                ******/
22:
23: #define         DXMIN   85      /* 128*512/768 */
24: #define         DXMAX   426     /* 640*512/768 */
25: #define         DYMIN   63
26: #define         DYMAX   479
27:
28: /****** パレット設定用定数経験的に値を決めている ******/
29: /******          赤と青は共通、緑は3の倍数        ******/
30:
31: #define         BR      24
32: #define         G       27
33: #define         G1      (G/3)
34: #define         G2      (G/3*2)
35: #define         G3      (G/3*3)
36:
37: /****** グローバル変数 ******/
38:
39: int             Dxf, Dxt, Dxt1, Dy, Xf1, Xf2, Xt1, Xt2, Y1, Y2;
40:
41: int             Xf[640];
42: unsigned short  Pf[512];
43: char            Pt[640], Pt1[640];
44: int             Buf[2][640];
45: unsigned char   RGB[480][80];
46: unsigned char   I[480][80];
47:
48: /****** 関数 ******/
49:
50: void    header();
51: void    conv();
52:
53: /****** プログラム ******/
54:
55: void    main( argc, argv )
56: int     argc;
57: char    *argv[];
58: {
59:         FILE                    *fp;
60:         int                     x1, y1, x2, y2;
61:         int                     ms_getdt, r, l, x, y, c, ms_curgt;
62:         unsigned char           filename[32];
63:         static struct TBOXPTR   tboxptr;
64:
65:         if ( argc<2 ) {
66:                 printf( "フルカラー画面をドット縦横比 1:1 の 16色に落とし、¥n" );
67:                 printf( "SX-WINDOW の キャンバス.x フォーマットのファイルにします。¥n" );
68:                 printf( "使い方 : sxconv [filename]¥n" );
69:                 printf( "       ファイル名に拡張子 (.PIX) はいりません。¥n" );
70:                 printf( "       矩形領域をマウスで指定して下さい。¥n" );
71:                 return;
72:         }
73:         if ( CRTMOD( -1 )!=12 ) {
74:                 printf( "スクリーンモードが違います。¥n" );
75:                 printf( "SCREEN 1,3,1 として画像をロードしておいてから実行して下さい。¥n" );
76:                 return;
77:         }
78:         MS_INIT();      /* マウスなどの初期設定 */
79:         SKEY_MOD(0,0,0);
80:         MS_CURST(0,0);
81:         MS_CURON();
82:         x1=y1=-1;       /* 座標値が負の場合に「座標未指定」というフラグも兼ねる */
83:         for (;;) {      /* 座標指定のループ開始 */
84:                 ms_getdt=(MS_GETDT()&0xFFFF);   /* マウスのボタンは押されたか */
85:                 r=((ms_getdt&0x00FF)==0x00FF);  /* 右ボタン */
86:                 l=((ms_getdt&0xFF00)==0xFF00);  /* 左ボタン */
87:                 ms_curgt=MS_CURGT();            /* マウスカーソルの座標を得る */
88:                 if ( (x1>-1) && (y1>-1) ) {
89:                         tboxptr.vram_page=1;    /* 領域を示すボックスを消す */
90:                         tboxptr.x=x1;
91:                         tboxptr.y=y1;
```

```
 92:                            tboxptr.x1=x-x1;
 93:                            tboxptr.y1=y-y1;
 94:                            tboxptr.line_style=0x0000;
 95:                            TXBOX( &tboxptr );
 96:                    }
 97:                    x=((ms_curgt&0xFFFF0000)>>16);  /* 新しいマウスカーソルのx座標 */
 98:                    y=(ms_curgt&0x0000FFFF);        /* 新しいマウスカーソルのy座標 */
 99:                    if ( (x1>-1) && (y1>-1) ) {     /* 強制的に画像サイズの範囲を制限する */
100:                            if ( x<(x1+DXMIN) ) x=x1+DXMIN;
101:                            if ( x>(x1+DXMAX) ) x=x1+DXMAX;
102:                            if ( y<(y1+DYMIN) ) y=y1+DYMIN;
103:                            if ( y>(y1+DYMAX) ) y=y1+DYMAX;
104:                            if ( (!r) && (!l) ) {
105:                                    tboxptr.vram_page=1;    /* 領域を示すボックスを描く */
106:                                    tboxptr.x=x1;
107:                                    tboxptr.y=y1;
108:                                    tboxptr.x1=x-x1;
109:                                    tboxptr.y1=y-y1;
110:                                    tboxptr.line_style=0xAAAA;
111:                                    TXBOX( &tboxptr );
112:                            }
113:                    }
114:                    if ( r ) {                      /* 右クリックでキャンセルする */
115:                            while ( (MS_GETDT()&0xFFFF) );  /* ボタンを離すのを待つ */
116:                            x1=y1=-1;
117:                    }
118:                    if ( l ) {                      /* 左クリックでポイントする */
119:                            while ( (MS_GETDT()&0xFFFF) );  /* ボタンを離すのを待つ */
120:                            if ( (x1>-1) && (y1>-1) && (x>x1) && (y>y1) ) {
121:                                    x2=x; y2=y;             /* 終点の座標 */
122:                                    break;                  /* ループを脱出して本処理に入る */
123:                            } else {
124:                                    x1=x; y1=y;             /* 始点の座標 */
125:                            }
126:                    }
127:            }
128:            MS_CUROF();     /* マウスなどを元に戻す */
129:            SKEY_MOD(-1,0,0);
130:
131:            strcpy( filename, argv[1] );
132:            strcat( filename, ".PIX" );
133:            if ( (fp=fopen( filename, "wb" ))==(FILE *)NULL ) {
134:                    printf( "ファイル'%s'がオープンできませんでした¥n", filename );
135:                    return;
136:            }
137:            Xf1=x1; Xf2=x2;
138:            Xt1=0;  Xt2=(x2-x1)*768/512;
139:            Y1=y1;  Y2=y2;
140:            header( fp );
141:            conv( fp );
142:            fclose( fp );
143:
144:            return;
145: }
146:
147: void    header( fp )
148: FILE    *fp;
149: {
150:            int             xf, xt, ex;
151:            int             i, size;
152:            unsigned short  pal[16];        /* ヘッダに書くパレット部 */
153:
154:            /******** 画像のサイズなどを調べる ********/
155:
156:            Dxt1=(Xt2-Xt1)+1;       /* ファイルには横16ドット単位で書き込む、そのサイズ */
157:            while( (Dxt1%16)!=0 ) Dxt1++;
158:
159:            Dxf = Xf2-Xf1;
160:            Dxt = Xt2-Xt1;
161:            Dy  = Y2-Y1;
162:
163:            for ( i=0; i<Dxt1; i++ ) Pt[i]=Pt1[i]=0;
164:
165:            /******** Xf[xt]を計算する(横方向拡大用) ***********/
166:
167:            xf = 0;
168:            ex = 2*Dxf-Dxt;
169:            for ( xt=0; xt<=Dxt; xt++ ){
170:                    Xf[xt] = xf;
171:                    while( ex>=0 ){
172:                            xf++;
173:                            ex -= 2*Dxt;
174:                    }
175:                    ex += 2*Dxf;
176:            }
177:
178:            /******** ヘッダを書く ********/
179:
180:            putc( 'T', fp );                /* キャンバス.x 用のID */
181:            putc( 'X', fp );
182:            putc( '1', fp );
183:            putc( '6', fp );
184:            size=(Dxt1)/8*(Dy+1)*4+40;      /* ファイルサイズ-8 */
185:            putc( (size>>24)&255, fp );
186:            putc( (size>>16)&255, fp );
187:            putc( (size>>8 )&255, fp );
188:            putc( (size    )&255, fp );     /* ここまでで8バイト書いているのでファイルサイズ-8 */
189:            putc( 0, fp );                  /* ? */
190:            putc( 0, fp );
191:            putc( 0, fp );
192:            putc( 0, fp );
193:            putc( (Dxt+1)/256, fp );        /* 画像のサイズ */
194:            putc( (Dxt+1)%256, fp );
195:            putc( (Dy+1)/256, fp );
196:            putc( (Dy+1)%256, fp );
```

▶いままでマニュアルが読みにくいとか文句たらたら言いながらプログラムを組んでました。しかし，いざ自分がドキュメントマニュアルを書いてみるとその言い尽くせない苦労の多さに気がつきました。もう文句は言いません。　小林　丈明（22）神奈川県

```
197:            pal[ 0] = rgb(  0,  0,  0 );    /* パレット設定 */
198:            pal[ 1] = rgb(  0,  0, BR );    /* 第1プレーンは青 */
199:            pal[ 2] = rgb( BR,  0,  0 );    /* 第2プレーンは赤 */
200:            pal[ 3] = rgb( BR,  0, BR );    /* 第3プレーンは緑その1 */
201:            pal[ 4] = rgb(  0, G1,  0 );    /* 第4プレーンは緑その2 */
202:            pal[ 5] = rgb(  0, G1, BR );
203:            pal[ 6] = rgb( BR, G1,  0 );
204:            pal[ 7] = rgb( BR, G1, BR );
205:            pal[ 8] = rgb(  0, G2,  0 );
206:            pal[ 9] = rgb(  0, G2, BR );
207:            pal[10] = rgb( BR, G2,  0 );
208:            pal[11] = rgb( BR, G2, BR );
209:            pal[12] = rgb(  0, G3,  0 );
210:            pal[13] = rgb(  0, G3, BR );
211:            pal[14] = rgb( BR, G3,  0 );
212:            pal[15] = rgb( BR, G3, BR );
213:            fwrite( pal, sizeof(short), 16, fp );   /* パレット書き込み */
214:
215:            return;
216: }
217:
218: /*******      第1(青)、第2(赤)のプレーンを書き込む       ********/
219: /*******    第3,4プレーン(緑)を書き込む…緑は2ビット      ********/
220: /******* 色調落としのアルゴリズムは1988年11月号p.66の ********/
221: /******* 桑野氏の記事を参照のこと(少し手抜きしてます) ********/
222:
223: void       conv( fp )
224: FILE       *fp;
225: {
226:            int       xt, y, i, p, b, carry, lc, lb;
227:
228:            for ( i=0; i<3; i++ ) {
229:                    for ( y=0; y<2; y++ ) {
230:                            for ( xt=0; xt<=Dxt; xt++ ) {
231:                                    Buf[y][xt]=0;
232:                            }
233:                    }
234:                    for ( y=Y1; y<=Y2; y++ ) {
235:                            get( Xf1, y, Xf2, y, Pf, (Dxf+1)*sizeof(short) );
236:                            lc=y&1;
237:                            lb=(y+1)&1;
238:                            carry=0;
239:                            for ( xt=0; xt<=Dxt; xt++ ) Buf[lb][xt]=0;
240:                            for ( xt=0; xt<=Dxt; xt++ ) {
241:                                    p = Pf[Xf[xt]];         /* ここで横方向拡大をしている */
242:                                    b = ( ( p>>(i*5+1) )&0x1F )*256+Buf[lc][xt]+carry;
243:                                    Pt[xt] = Pt1[xt] = 0;
244:                                    switch ( i ) {
245:                                    case 0:         /* 青 */
246:                                    case 1:         /* 赤 */
247:                                            if ( b>=31*256 ) {
248:                                                    b -= (31*256);
249:                                                    Pt[xt] = 1;
250:                                            }
251:                                            break;
252:                                    case 2:         /* 緑 */
253:                                            if ( b>=30*256 ) {
254:                                                    b -= (31*256);
255:                                                    Pt[xt]  = Pt1[xt] = 1;
256:                                            } else {
257:                                                    if ( b>=20*256 ) {
258:                                                            b -= (20*256);
259:                                                            Pt1[xt] = 1;
260:                                                    } else  {
261:                                                            if ( b>=10*256 ) {
262:                                                                    b -= (10*256);
263:                                                                    Pt[xt] = 1;
264:                                                            }
265:                                                    }
266:                                            }
267:                                            break;
268:                                    }
269:                                    carry = b/8;
270:                                    Buf[lb][xt] += carry;
271:                                    if ( xt>0 ) {
272:                                            Buf[lb][xt-1] += (carry*2);
273:                                    } else {
274:                                            Buf[lb][xt  ] += (carry*2);
275:                                    }
276:                                    if ( xt<Dxt ) {
277:                                            Buf[lb][xt+1] += carry;
278:                                    } else {
279:                                            Buf[lb][xt  ] += carry;
280:                                    }
281:                                    carry *= 4;
282:                            }               /* 1ラインぶんをまとめてファイルに書き出す */
283:                                            /* このとき垂直型→水平型の変換を行う */
284:                            for ( xt=0; xt<Dxt1; xt+=16 ) {
285:                                    RGB[y-Y1][xt/8]=((Pt[xt]<<7)|(Pt[xt+1]<<6)|(Pt[xt+2]<<5)|(Pt[x
t+3]<<4)|(Pt[xt+4]<<3)|(Pt[xt+5]<<2)|(Pt[xt+6]<<1)|(Pt[xt+7]));
286:                                    RGB[y-Y1][xt/8+1]=((Pt[xt+8]<<7)|(Pt[xt+9]<<6)|(Pt[xt+10]<<5)|
(Pt[xt+11]<<4)|(Pt[xt+12]<<3)|(Pt[xt+13]<<2)|(Pt[xt+14]<<1)|(Pt[xt+15]));
287:                                    if ( i<2 ) continue;
288:                                    I[y-Y1][xt/8]=((Pt1[xt]<<7)|(Pt1[xt+1]<<6)|(Pt1[xt+2]<<5)|(Pt1
[xt+3]<<4)|(Pt1[xt+4]<<3)|(Pt1[xt+5]<<2)|(Pt1[xt+6]<<1)|Pt1[xt+7]);
289:                                    I[y-Y1][xt/8+1]=((Pt1[xt+8]<<7)|(Pt1[xt+9]<<6)|(Pt1[xt+10]<<5)
|(Pt1[xt+11]<<4)|(Pt1[xt+12]<<3)|(Pt1[xt+13]<<2)|(Pt1[xt+14]<<1)|Pt1[xt+15]);
290:                            }
291:                    }
292:                    for ( y=0; y<=Dy; y++ ) fwrite( RGB[y], Dxt1/8, 1, fp );
293:                    if ( i<2 ) continue;
294:                    for ( y=0; y<=Dy; y++ ) fwrite( I[y], Dxt1/8, 1, fp );
295:            }
296:            return;
297: }
```

▶えーと，だからディスク内蔵のOh！X SUPER-HDと従来どおりのOh！X EXPERT，低価格バージョンのOh！X PROとかにしてくれれば読者に選択の余地ができていいんじゃないかな，とかって……。
及川　一志（20）宮城県

BASIC

パズルゲームを作る(前)

Izumi Daisuke 泉 大介

X-BASICでは遅くて本格的なゲームは作れないと思っている読者もいるかもしれません。ところがどっこい，いまOh!X編集部でいちばん流行っているゲームが，これ「YET」。プレイしてから読むか，読んでからプレイするか。君はどっちだ?

前回のエレベータシミュレーションはいかがだったでしょうか。ちょっと堅い内容ではありましたが，コンピュータによるシミュレーションの雰囲気を感じていただけたのではないかと思います。さて今回は気分を一新して，ゲーム作りに取り組んでみたいと思います。作成するゲームは今月の付録ディスクに収めてあるYETです。どんなゲームか，特集記事を読んでください。BASICのスピードの問題もあり，まったく同じものを作るというわけにはいきませんが，細かい凝った部分を取り除いたシンプルなYETをお届けしたいと思います。

タイル表示方法のいろいろ

みなさんは画面に文字が表示される仕組みをご存じでしょうか。画面上に表示される英数字は，8×16のドットの集まりで表現されています。このドットの長方形の中の特定のものを光らせることによって文字を表現しているのです。どのドットを光らせれば「A」という文字が表示できるのかという情報を収めてあるのがCGROMです。光らせるドットを1，光らせないドットを0で表現することにすると，英数字1文字は8個の0，1の並びを16組使って表すことができる，ということになります。8個の0，1の並びは8桁の2進数とみなすことができます。8桁の2進数のことを1バイトといいますから，英数字1文字は16バイトのデータで表現されている，ということもできるわけです。16バイトのデータをCGROMから読み出し，それを文字を表示するために用意された特定のRAM (VRAMと呼ばれる)にコピーする。こうしてやっと1文字表示できるのがX68000のテキスト画面です。

●文字を使う

X68000以外の多くのコンピュータでも，文字が表示される原理は同じです。ただし，VRAMに書き込むのは文字のASCIIコードだけでいいようになっています。CGROMからデータを読み出して画面に文字として表示する部分はハードウェアが自動的に行ってくれるのです。ASCIIコードは0～255で，これは1バイトのデータが表現できる範囲と同じです。つまり，X68000が16バイトのデータ転送を行う必要があるところを1バイトですませられるようになっているわけです。これはZ80などのCPUパワーの低いマシンで，1文字表示するごとに16バイトのデータ転送をCPUが行っていたのでは遅くて使いものにならないからです。X68000のやり方は，データ転送時にデータをいじれば簡単に太文字にしたり任意の大きさに拡大することができる，自由度の高い方法なのです。

とはいうものの，YETのように多くのタイルが動くものを文字を使って表現するのはやはり苦手です。ひとつのタイルを全角文字で表したとすると，タイルをひとつ動かすごとに32バイトのデータ転送が必要になるのですから大変です。ほかのパソコンのピコピコゲームでは敵や自機を文字で表現するということが普通に行われていますが，X68000にとってはうまいやり方とはいえないでしょう。

●スプライトを使う

X68000は敵や自機などのキャラクターを表示するのにもってこいのスプライトというものを持っています。65536色中の16色を使って表現したキャラクターを，スムーズに動かすことができるスプライトは，これまでにも何度か取り上げましたね。スプライトパターンを画面に表示する部分はハードウェアがやってくれますから文字よりは速そうですし，色をつけられるという利点まであります。ただし，スプライトは1画面に128個までしか表示できません。YETは最大で23×9=207個のタイルが表示される可能性がありますから，これでは不十分です。さらに，色が揃ったタイルは画面から消去されますので，スプライトのどのプレーンが空いたのかを保持しなければならず面倒そうです。

もうひとつのスプライトBG

そこで登場するのがBGです。これはスプライトの一種ですが，スプライトとは異なり画面の任意の位置に表示することはできません。文字のように[1]決まった位置にしか表示できないのです。512×512ドットのモードで使用しているときには，縦横ともに16ドットおきになります。つまり，画面に漢字を表示するのと同じようなものです。ただしこの「漢字」は任意のパターンを定義することができ，しかもそれぞれのドットに色まで着けることができます。スプライトと異なるもうひとつの点は，任意のひとつをスムーズに移動させることができないというこ

1) もちろんこれはBASICなどを標準的に使っている場合の話です。マシン語を使ってモリモリプログラムすれば，1ドット単位で任意の位置に表示することができます。

とです。BGでは全画面をスムーズにスクロールさせることしかできません。おそらくこのためBG(Back Ground)という名前がついたのでしょう。すなわちゲームなどで使う「背景」を表示しておき，それをスムーズにスクロールさせて奥行き感を出すのに使うために用意されたもの。そんなところでしょうか。

●BGの使い方

まずは16×16のパターンを定義しなければなりません。BGもスプライトですから，これはsp_def関数で行います。また当然ながら，表示するにはsp_disp関数を使います。初期化はsp_initです。どれもスプライトのときに使った関数ばかりですね。もちろんパターン作成にはDEFSPTOOL.BASが使えます。

こうして作ったパターンをスプライトとして使うならsp_set，sp_move関数を使うのですが，BGの場合はここからが異なっています。

BGは2つのプレーンを持っていて，表示したいパターンはこのプレーンに書き込みます。このための関数がbg_putです。そして2つのプレーンのうち，画面に表示するほうを指定するのがbg_set関数です。256×256ドットモードでは2プレーンを同時に表示することができますが，512×512ドットモードでは1プレーンしか表示できません。マニュアルではBGを書き込むプレーンのことを「テキストページ」と呼んでいますので，以後はこれにならうことにします。

では実際にやってみましょう。リスト1です。320行以降のsprite_pattern関数は，例によってDEFSPTOOL.BASで作成したものです。ここでは2つのパターンが定義されています。最初のものはタイル用，2番目のものはレンガ用です。90行からのsprite_color関数では，タイルで使う色を設定しています。ここはあとで詳しく説明しましょう。

さてメインルーチンですが，まずは10行で512×512×65536色モードにし，続いてスプライトの初期化，スプライトパターンの設定，色設定，そしてスプライトの表示を順に指示しています。ここまでは以前やったときと同じですね。60行のbg_set関数が，問題のBG用の設定をやっている関数です。

256×256ドットモードではテキストページを2つとも表示できるといいました。どちらのテキストページを上に表示するかを示しているのがbg_setの最初のパラメータです。マニュアルでは「バックグラウンド0」「バックグラウンド1」と表記されています。バックグラウンド0に指定したテキストページのほうが上に表示されます。ただし512ドットモードではバックグラウンド0しか使えません。したがってバックグラウンド0に割り当てたテキストページだけが表示されることになり，「512ドットモードではBGはひとつ」ということになるのです。

リスト1を入力したらrunしてみてください。画面モードが変わっただけですぐに終了しましたね。現在60行のbg_set命令によって，テキストページ1がBGとして画面に表示されています。定義したパターンをテキストページ1に書き込んでみましょう。

bg_put(1, 10, 10, &B01000000001)

で青いタイルが表示されたらOKです。

bg_putの最初のパラメータはパターンを書き込むテキストページの番号です。続く2つのパラメータは表示する座標を表しています。512ドットモードではBGは16ドットおきですから，これはグラフィック座標でいえば(160,160)と同じ場所になります。最後のパラメータは「使用するパレットブロック」と「表示するパターン番号」を2進数で表したもの

リスト1
BGのテストプログラム

```
 10 screen 1,3,1,1                    /* 512×512×65536色
 20 sp_init()                         /* スプライトの初期化
 30 sprite_pattern()                  /* パターンの設定
 40 sprite_color()                    /* パレットの変更
 50 sp_disp( 1 )                      /* スプライト表示
 60 bg_set( 0, 1, 1 )                 /* ページ1を表示
 70 end
 80 /*
 90 func sprite_color()
100   int i, j
110   int r, g, b
120   int cr, cg, cb                  /* Color of Red … …
130   for i=1 to 6                  /* 2進数でいうと001～110
140     r = i / 4                     /* 1・・ を判定
150     g = (i mod 4) / 2             /* ・1・ を判定
160     b = i mod 2                   /* ・・1 を判定
170     for j=0 to 4
180       cr = r * ( j*4 + 15 )       /* jに応じて
190       cg = g * ( j*4 + 15 )       /* 明るさを
200       cb = b * ( j*4 + 15 )       /* 設定
210       sp_color( j+1, rgb( cr, cg, cb ), i+1 )
220     next
230     sp_color( 6, 1, i+1 )           /* 6は黒
240   next
250                                   /* 枠の色作り
260   for i=0 to 2
270     sp_color( i+1, hsv(16,31,i*2+12 ), 8 )
280   next
290   sp_color( 4, hsv(22,31,22), 8 )
300 endfunc
310 /*
320 func sprite_pattern()
330   dim char c(255)
340   int i
350   c={
360     +15,15,15,15,15,15,15,15,15,15,15,15,15,15,15,15,
370     +15, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,14,
380     +15, 6, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6,14,
390     +15, 6, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 6,14,
400     +15, 6, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 6,14,
410     +15, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6,14,
420     +15, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6,14,
430     +15, 6, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6,14,
440     +15, 6, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6,14,
450     +15, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6,14,
460     +15, 6, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 6,14,
470     +15, 6, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 6,14,
480     +15, 6, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 6,14,
490     +15, 6, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6,14,
500     +15, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,14,
510     +14,14,14,14,14,14,14,14,14,14,14,14,14,14,14,14
520   }
530   sp_def(1,c)
540   c = {
550     +3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
560     +3, 4, 4, 4, 4, 4, 4, 1, 3, 4, 4, 4, 4, 4, 4, 1,
570     +3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1,
580     +3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1,
590     +3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1,
600     +3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1,
610     +3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1,
620     +3, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1,
630     +3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
640     +4, 4, 4, 1, 3, 4, 4, 4, 4, 4, 4, 1, 3, 4, 4, 4,
650     +2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2,
660     +2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2,
670     +2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2,
680     +2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2,
690     +2, 2, 2, 1, 3, 4, 2, 2, 2, 2, 2, 1, 3, 4, 2, 2,
700     +1, 1, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1, 3, 1, 1, 1
710   }
720   sp_def(2,c)
730 endfunc
```

です。マニュアルには，

ビット８～11……パレットブロック

ビット０～７……パターンコード

と書いてあります。これはパラメータを２進数で表現して説明しているのです。２進数はいちばん右の桁をビット０といいますから，「パレットブロック１，パターンコード１」なら，

00100000001

になるんだよ，と書いてあるわけです。これでは使いづらいから，次のような実験をしてみましょう。

```
print &B00100000000
```

&Bは後ろに続く数が２進数であることを表しています。これはパレットブロック１のパターン０を指示する２進数です。続いて，

```
print &B01000000000
```

も試してみてください。これはパレットブロック２のパターン０を指示しています。どうですか？　最初のは256，次のは512になりましたね。そう，256の倍数になっているのです。つまり，パレットブロックを256倍し，それにパターンの番号を加えてやれば目的のパラメータを作り出すことができるわけです。

ではいろいろ試してみましょう。まだ説明はしていませんがsprite_color関数ではパレットブロック２～８の色を設定しています。このうち２～７はタイルの色，８はレンガの色をセットしています。

```
bg_put(1,20,20,256×5+1)
bg_put(1,10,10,256×8+2)
```

などとして，遊んでみてください。タイルは１種類のパターンを色を変えて表示することで６種類を作り出しています。

タイルの表示方法

YETではタイルのパターンをひとつだけ用意し，異なるパレットブロックを使って表示することで６種類のタイルを作っています。sp_init関数を実行すると，すべてのパレットブロックは初期化されデフォルトの16色がセットされます。そこでパレットブロックに手を加え，美しい(！)グラデーションを作っているのがsprite_color関数です。

この関数では青，緑，シアン，赤，マゼンタ，黄の６つの色のグラデーションを作り出しています。その方法は２進数を利用したちょっと複雑な方法ですが，原理さえわかってしまえば簡単なものです。

130行のfor～nextで，変数iの内容は１～６まで変わっていきます。これを２進数で表現すると，

001，010，011，100，101，110

となります。140～160行はこれに応じて，

1)　３桁目が１ならrを１に

2)　２桁目が１ならgを１に

3)　１桁目が１ならbを１に

しているところです[2)]。もし該当桁が０ならば，対応するr，g，bも０になります。170～220行では，

j×4+15

という計算によって，jの値に応じて大きくなっていく数を作り出し，それをrgb関数に与える色成分としてcr，cg，cbにセットしています。そして210行のsp_color関数で色を変更するのです。

具体的に追いかけてみましょう。まずiが１のときです。１桁目が１ですからbだけが１となり，rとgは０になります。したがってcr，cgも０のままで，変化するのはcbだけです。170行のfor～nextループで，最初jは０ですからcbは15，jが１のときには19，続いて23，27，31と変わっていきます。ですから210行は，

```
sp_color(1, rgb(0, 0, 15), 2)
sp_color(2, rgb(0, 0, 19), 2)
sp_color(3, rgb(0, 0, 23), 2)
sp_color(4, rgb(0, 0, 27), 2)
sp_color(5, rgb(0, 0, 31), 2)
```

とやっているのと同じことになります。こうしてパレットブロック２の１～５の色が，次第に明るくなっていく青になるわけです。ほかの色も同様にしてセットされます。

これでタイルとレンガの用意はできました。次にタイルを動かす方法を考えてみましょう。

タイルを落とす

ちょっと次のプログラムを実行してみてください。

```
for i=0 to 30
  locate i×2, 10
  print " "
  locate i×2+2, 10
  print "→"
next
```

コロンで区切って１行に書いて実行してください。「→」が右に飛んでいくように見えたでしょう。

プログラムはまず座標(i,10)に表示してあるものをスペースで消し，続いてその右隣りに「→」を表示しているだけです。iがひとつ大きくなると，前回描いた「→」はスペースで消されさらにその右に「→」が表示されます。こうして前に描いたものを消し，新しい位置にキャラクターを描き直すだけで人間にはそれが動いているように見えるのです。同じ手法はBGにも使えます。リスト１を実行したあと，

```
for i=0 to 30
  bg_put(1, i, 10, 256)
  bg_put(1, i+1, 10, 256×2+1)
  for j=0 to 100 : next
next
```

というプログラムを入力し実行してみてください(コロンで区切って１行に書くこと)。あまりに速いので間に空ループを入れてあります。中のjのループを除けばBGがいかに速いかわかっていただけるでしょう。スプライトは新しい表示位置を指定するだけで動かすことができましたが，文字やBGを動かすにはこのような方法を使います。

2)　ビット演算を使うという方法もありますが，まだ説明していませんのでまたの機会にしましょう。

そしてタイルが落ちるまで

BGを動かす方法はおわかりいただけましたね。いまは横向きに動かしましたが，縦向きに動かす場合も同じです。前に描いたものを消し，新しい位置に描き直せばいいのです。YETのプログラムを作るうえでもうひとつ考えておかなければならないことは，タイルが落ちてくるときに必要になるウエイトです。先ほど見たようにウエイトなしではあっという間に移動してしまいますからね[3]。

先の例ではfor～nextの空ループを使ってウエイトを入れていました。これはよく使う手法なのですがひとつ問題点があります。コンパイルすると速度が変わってしまうのです。コンパイルしたあとでは，たかだか100回程度の空ループなどあってもなくても変わらないのです。コンパイル後の速度を見てループ回数を調整してもいいのですが，ここではFM音源を使うという方法を採りました。ここで作るYETはインタプリタ上で実行することを前提に作っていますが，コンパイルするプログラムを作るときのひとつの解決策として利用してみてください。

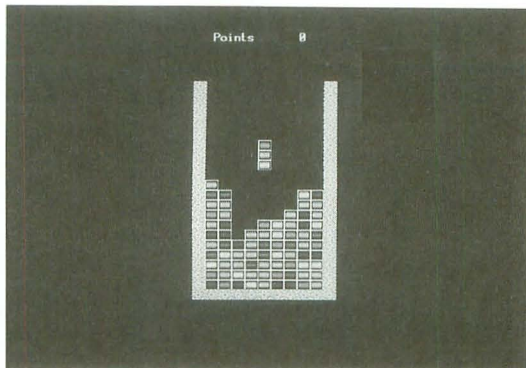
プログラムにいく前に，入力してあるリスト1をリナンバしておきましょう。10～70行は必要ありませんので，まず「delete 10-70」として削除してしまってください。そして「renum 2000」で2000行からに行番号をつけ直します。

ではリスト2です。関数から見ていきましょう。リストの最後，1130行にあるinitFM関数はFM音源でウェイトを入れるための前処理を行っています。まずトラックをひとつ確保し，そこに長い休符をセットします。休符ですからもちろん音は鳴りません。指定したトラックが演奏中かどうかを調べるm_stat関数を使って，

```
m_play(1)
repeat : until m_stat(1) = 0
```

とすれば「休符を演奏し終わる」までrepeat～untilループを回り続けますから，これでウエイトをかけることができるのです。ディスクに収録したYETのように徐々にスピードを速くしたい（ウエイトを短くしたい）なら，長さの違う休符をいくつか別のトラックに確保しておいて，条件に応じてどれを鳴らすかを決めるだけです。この実例は来月紹介します。

970行のmakeScreen関数は，ゲーム盤を作る関数です。まず990行で暗いグリーンで画面を塗ります。1000行でテキストページ1を256(パレットブロック1，パターン0）で埋め，1010～1070行でレンガのパターンを使って枠を表示します。そして，画面の上のほうに「Points 0」と表示し，終了します。今月はまだポイントを計算するところは作りませんが，縁起ものということで入れておきました。

さてお次は10～70行の変数宣言です。タイルは縦に3つ連なって落ちてきます。このうちいちばん上

のタイルが表示されている座標を入れてあるのがtileX，tileYの2つの変数です。3連のタイルはtiles配列に収められています。ここには，

tiles(0)＝256×3＋1

のように，パターンデータがそのまま収められます。

30行のgameOverは，ゲームが終わったかどうかを保持するフラグ，40行のkeySenはキーボード入力を受け付けるかどうかを保持するフラグです。「2」やスペースでタイルを落とした場合には以後のキー入力を無効にするため用意しました。50～70行はメインルーチンで使う変数です。

メインルーチンに入る前に，もうひとつ関数の説明をすませておきましょう。830行のmoveTile関数です。これは3連のタイルを横に移動させるために用意しました。850～870行で移動させようとしているx座標に，タイルやレンガがないかどうかを調べます。makeScreen関数でテキストページ1を256で埋めましたね。したがってテキストページ1を調べ，256以外のものがあればそれはタイルかレンガだと判断できます。このときは移動できないということで即座にリターンします。なければ880～900行でいま表示されている3連のタイルを消し，910～940行で新しいx座標に移動させます。

最後にお待ちかねのメインルーチンです。180行のwhileは無限ループになっていて，ゲームオーバーになるまで新しい3連のタイルを発生させます。実際にタイルを作っているのは240～320行のfor～nextループです。ここで，発生させようとしている位置にすでにタイルがあるかどうかも見張っています。もしあればゲームオーバー（タイルが上まで積まれた）です。タイルは256×2＋1～256×7＋1ですから，

int(rnd(　)×6＋2)

で2～7を発生させ，これを256倍して1を加えればOKですね。これをやっているのが300行です。このfor～nextループでは，tiles配列へのセットと画面への表示を300，310行で同時にやっています。

350行でトラック1（長い休符）を鳴らし，続いて360行ではキーボードが押されなくなるのを待ちます。そして360行のwhileでタイルの落下開始です。

タイルが落下している間にやらなければならないことは，キー入力に応じてタイルを左右に動かしたり，3連のタイルを回転させることです。360行のキ

3） 実際には3つのタイルを一度に動かしたり，下になにもないかなどの判定がありますので，ここで見た例ほど速く落ちるわけではありません。しかしそれでも，十分なにもできない速度ではあります。

ー入力を受けてこれらの処理を行うのが390～570行のswitch文です。ここではinkey$(0)で読み込んだ文字のASCIIコードで比較を行っています。もちろん,

```
switch ( ch )
  case "4"
    ......
  case "6"
```

などとしてもOKなのですが，これではコンパイルできないのです。自作プログラムをコンパイルする人はご用心を。

4，6のキーが押されたときは左右に動かすだけですから処理は簡単。moveTile関数を呼び出すだけで終了です。5を押した場合はタイルが回転します。どのように回転させているのか，じっくり追いかけてみてください。代入と表示が交互になっているだけですから，分けて追いかけると簡単かもしれません。最後がスペースまたは2が押されたときです。このときはkeySenフラグを0にし，380行でキー入力ができないようにします。

これでキー入力とそれに付随する処理はおしまいです。650～760行はタイルをひとつ下に下げる処理をやっています。この処理は次の2つの条件のどちらかが成立したときに行います。

1) 休符が鳴り終わった（ウエイト終了）
2) 2またはスペースが押された

これを判定しているのが650行のif文です。660行で3連のタイルの下になにもないか調べ，なければ670～730行でタイルを下に移動させます。そして750行で再び休符を鳴らします[4]。

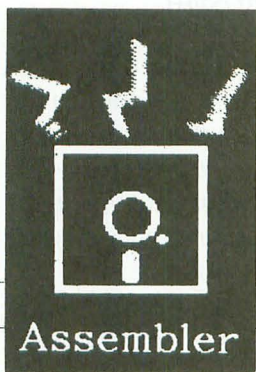
770行で3連のタイルの下を調べ,なにもなければループを継続します。なにかあれば，落下処理は終了します。そしてこのwhileループを抜け，180行に戻って新しいタイルの生成から再び始まることになります。

＊

今回はBGの使い方とYETプログラムの前半部分をやりました。来月はYETプログラムの後半をお届けします。YETは同じ色のタイルを縦，横，斜めに3つ以上揃えると揃ったタイルが消えます。縦，横，斜めに3つ以上並んだかどうか，どうやって確かめればいいのでしょう。考えてみてください。

4) スペースまたは2が押されたときは，このm_playにはなんの意味もありません。

リスト2 YETその1

```
  10 int tileX, tileY                  /* タイルの座標
  20 int tiles( 2 )                    /* 落下中のタイル
  30 int gameOver
  40 int keySen                        /* キー入力フラグ
  50 int i
  60 str ch
  70 int tmp                           /* タイル待避用
  80 /*
  90 screen 1,3,1,1                    /* 512×512×65536色
 100 sp_init()                         /* スプライトの初期化
 110 sprite_pattern()                  /* パターンの設定
 120 sprite_color()                    /* パレットの変更
 130 sp_disp( 1 )                      /* スプライト表示
 140 bg_set( 0, 1, 1 )                 /* ページ1を表示
 150 initFM()                          /* FM音源の設定
 160 makeScreen()                      /* 盤面作り
 170 /*
 180 while 1
 190   tileX = 16                      /* 最初の位置
 200   tileY = 3
 210   keySen = 1                      /* キー入力可
 220   /*
 230   gameOver = 0                    /* ゲーム続行
 240   for i=0 to 2
 250     if bg_get( 1, tileX, tileY+i ) <> 256 then {
 260       gameOver = 1  /* 表示位置にタイルがあれば終わり
 270       break
 280     }
 290                                   /* 新しいタイル生成
 300     tiles( i ) = int( rnd()*6 + 2 )*256 + 1
 310     bg_put( 1, tileX, tileY+i, tiles(i) )
 320   next
 330   if gameOver then break          /* whileを抜ける
 340   /*
 350   m_play( 1 )                     /* ウェイト開始
 360   while inkey$(0)<>"" : endwhile /* 先行入力をクリア
 370   while 1                             /* 落下開始
 380     if ( keySen ) then ch = inkey$(0) /* キー入力
 390     switch ( asc(ch) )
 400       case '4'                    /* 右へ移動
 410         moveTile( tileX-1 ) : break
 420       case '6'                    /* 左へ移動
 430         moveTile( tileX+1 ) : break
 440       case '5'                    /* タイル回転
 450         tmp = tiles( 2 )
 460         bg_put( 1, tileX, tileY+2, tiles( 1 ))
 470         tiles( 2 ) = tiles( 1 )
 480         bg_put( 1, tileX, tileY+1, tiles( 0 ))
 490         tiles( 1 ) = tiles( 0 )
 500         bg_put( 1, tileX, tileY, tmp )
 510         tiles( 0 ) = tmp
 520         break
 530       case ' '                    /* タイルを落とす
 540       case '2'
 550         keySen = 0
 560         break
 570     endswitch
 580     if ch = chr$(27) then {       /* 一時停止
 590       while inkey$(0)<>"" : endwhile
 600       while inkey$(0)="" : endwhile
 610     }
 620     /*
 630     /* ウェイト終了、またはタイル落下の処理
 640     /*
 650     if m_stat( 1 )=0 or keySen=0 then {
 660       if bg_get( 1, tileX, tileY+3 ) = 256 then {
 670         for i=0 to 2  /*下にタイルがなければ1つ下へ
 680           bg_put( 1, tileX, tileY+i, 256 )
 690         next
 700         tileY = tileY + 1
 710         for i=0 to 2
 720           bg_put( 1, tileX, tileY+i, tiles(i) )
 730         next
 740       }
 750       m_play( 1 )               /* 再びウェイトをかける
 760     }
 770     if bg_get( 1, tileX, tileY+3 ) <> 256 then break
 780   endwhile
 790   if gameOver then break
 800 endwhile
 810 end
 820 /*
 830 func moveTile( newx )                /* タイルの横への移動
 840   int i
 850   for i=0 to 2                       /* 横に障害物はないか
 860     if bg_get( 1, newx, tileY+i ) <> 256 then return()
 870   next
 880   for i=0 to 2                       /* なければ移動
 890     bg_put( 1, tileX, tileY+i, 256 )
 900   next
 910   for i=0 to 2
 920     bg_put( 1, newx, tileY+i, tiles(i) )
 930   next
 940   tileX = newx                       /* tileXを更新
 950 endfunc
 960 /*
 970 func makeScreen()
 980   int i
 990   fill( 0, 0, 511, 511, rgb( 0, 16, 0 ))
1000   bg_fill( 1, 256 )                  /* BGを256で埋める
1010   for i=5 to 26                      /* レンガを縦に積む
1020     bg_put( 1, 11, i, 256*8+2 )
1030     bg_put( 1, 21, i, 256*8+2 )
1040   next
1050   for i=11 to 21                     /* レンガの床を作る
1060     bg_put( 1, i, 26, 256*8+2 )
1070   next
1080   color 7                            /* 白の太文字
1090   locate 25, 1
1100   print using "Points  ######";0 /* 消したタイル数
1110 endfunc
1120 /*
1130 func initFM()
1140   m_alloc( 1, 40 )
1150   m_trk( 1, "@L192R" )               /* 最長の音をセット
1160   m_assign( 1, 1 )
1170   m_tempo( 200 )
1180 endfunc
```

脱“入門編”のための身辺整理

Murata Toshiyuki 村田 敏幸

マシン語プログラミングも連載開始から1年を過ぎ，入門編はひと区切りということになりました。これからはマシン語ならではの特長をより生かすかたちでプログラミングに取り組む予定です。まずは入門編で触れなかった基本事項の総チェックからいきましょう。

えー，業務連絡。連載開始当初から読んでいただいた熱心な読者の方ならもう十分な基礎体力がついたころだし，1年も経っているのに“入門”でもないだろうということで，今回からタイトルの《入門編》をはずすことにした(ほら↑)。入門編では Human68kのDOSコールに絞って話を進めてきたわけだが，今後はそういう制限は一切なしのなんでもあり，でやっていきたい。といって，いきなり内容が高度になるわけじゃあないからビビらないよう，また，過度の期待をしないようお願いする。とりあえず，今後ともよろしく。

今回は入門編でははしょった部分をフォローして明日へつなぐという主旨で，こまごまとした話を書き連ねてみる。自慢じゃないが，話に一貫性は，ない。まず68000のアドレッシングモードの総ざらいが始まる。

アドレッシングモードの総まとめ

68000は豊富なアドレッシングモードを備えている。入門編ではなるべく複雑なアドレッシングモードを使うのを避けてきたが，それでもなんだかんだで以下の9種類のアドレッシングモードが登場している([ ]内は例)。

・データレジスタ直接形式 [move.l d1, d0]
・アドレスレジスタ直接形式 [move.l a0, d0]
・アドレスレジスタ間接形式 [move.l (a0), d0]
・ポストインクリメントアドレスレジスタ間接形式 [move.l (a0)+, d0]
・プリデクリメントアドレスレジスタ間接形式 [move.l −(a0), d0]
・ディスプレースメント付きアドレスレジスタ間接形式 [move.l 4(a0), d0]
・イミディエイトデータ形式 [move.l #1234, d0]
・クイックイミディエイトデータ形式 [moveq.l #10, d0]
・絶対ロングアドレス形式 [move.l LABEL, d0]

このうち，多少ややこしいのがアドレスレジスタ間接関係かもしれない。いまさらの感はあるが，念のため要点を確認しておこう。

1) レジスタ直接と間接の違い

アドレスレジスタ直接形式はアドレスレジスタの中身自体を対象にする。対して，アドレスレジスタ間接形式は“アドレスレジスタが指す(ポイントする)メモリ”を操作対象にする。

2) ポストインクリメント/プリデクリメントアドレスレジスタ間接形式とサイズの関係

ポストインクリメントアドレスレジスタ間接形式はアドレスレジスタが指すメモリを操作したあとでアドレスレジスタの値を増やし，プリデクリメント〜のほうはメモリアクセスに先立ってアドレスレジスタの値が減じられ，減らしたあとのアドレスレジスタによって指されるメモリが操作対象となる。ここで，アドレスレジスタが増減される量は演算のサイズに等しく，バイトのときは1，ワードのときは2，ロングワードのときは4だった。

3) ディスプレースメント付きの意味

ディスプレースメント付きアドレスレジスタ間接形式では“アドレスレジスタにディスプレースメントを加えたアドレスで指定されるメモリ”が操作対象になる。ディスプレースメントは16ビットの符号付き数として扱われるから，アドレスレジスタが指す位置の前後−32768〜+32767の任意のメモリを指定できることになる。ディスプレースメントをアドレスレジスタに加算して実効アドレスを求めるときには自動的に符号拡張が行われるのを忘れてはならない。a0に50000Hが入っているときに，

```
move.w  $8000(a0), d0
```

で参照されるのはa0に00008000Hを加えた58000H番地ではなく，FFFF8000Hを加えた(=00008000Hを引いた)48000Hとなる。

4) どんなときにどのアドレッシングモードを使うか

ポストインクリメント/プリデクリメントアドレスレジスタ間接形式は，スタックの操作やメモリ上の配列状に並べられたデータを次々に操作していくのに重宝する。また，ディスプレースメント付き〜はスタックフレームのアクセスや，構造を持ったデー

タを扱うのに便利だ。プログラミングするうえでは，ディスプレースメントは数値で直接指定するよりもequや.offsetなどを使って記号定数に定義して使うほうがわかりやすいという話もした。

なお，基本的にはアドレスレジスタ間接形式だけ知っていれば，ほかの長ったらしい名前のアドレッシングモードを使わずとも同等の処理を行うことができる。たとえば，

```
move.l (a0)+, d0
```

に相当する処理は，

```
move.l (a0), d0
addq.l #4, a0
```

の2命令で実現されるだろう。こういう理屈さえわかっていれば，あとはどうにでもなるものだ。

*

ここで，アドレスレジスタ間接アドレッシングの最後のバリエーション，“インデックス付きアドレスレジスタ間接形式”を紹介しておく。このアドレッシングモードは，

```
move.b 2(a0, d0), d1
```

のようにして使用し，一般形は，

```
disp8(an, index)
```

となる。“アドレスレジスタにインデックスと8ビットディスプレースメントを加えたアドレス”を指定するという少々複雑なアドレッシングモードだ。インデックスには任意のデータレジスタおよびアドレスレジスタが利用でき，しかもインデックスのロングワードすべてを使用するか，下位ワードだけを使用するかを指定することができる。インデックスのサイズがワードかロングワードかの指定は，

```
move.b 2(a0, d0.w)
move.b 2(a0, d0.l)
```

のように後ろに“.w”か“.l”を付けて表す。たとえば，

```
move.b 2(a0, d0)
```

のように省略するとワードと見なされる。また，ワードサイズのインデックスはアドレス計算時に32ビットに符号拡張される。

ディスプレースメントは8ビットの符号付き数として扱われる。そのため範囲は－128～＋127とちょっと狭めになっている。これもアドレス計算時には32ビットに符号拡張してから加算される。

インデックス付きアドレスレジスタ間接形式は多少複雑なデータ構造（Cでいうところの構造体の配列とか）を扱うときにはなかなかの威力を発揮する。また，あまり勧められる手法ではないが，

```
lea.l 4(a0, d0.l), a1
pea.l 4(a0, d0.l)
```

のように，leaやpeaと組み合わせると2つのレジスタ＋定数の和を1命令で求めるのにも使えてしまう。a0.l+d0.l+4がそれぞれa1かスタックトップに求まるわけだ。アドレスレジスタを本来のポインタとしてではなくデータを格納するのに使うあたりが邪道っぽくて気持ち悪い。

プログラムカウンタの効用

この連載の初期に，アドレスの概念を説明した際，任意のメモリは80000H番地というような絶対的なアドレスで指定する以外に，ある基準からの相対的な差でも表現できるという話をした。ディスプレースメント付きアドレスレジスタ間接形式なんかはアドレスレジスタを基準にして相対的にアドレスを指定するアドレッシングモードといえる。

では，アドレスレジスタ以外に基準に使えるものはないだろうかと考えていくと，プログラムカウンタがあることに気づく。PCはプログラムの実行に伴ってどんどん更新されてしまうから基準となり得ないように見えるかもしれない。が，ある命令を実行しようとしたその瞬間には，PCがその命令の置かれた位置自体をポイントしているのは間違いない。そのため“実行する命令が置かれた位置から4バイト先”というような形で特定のアドレスを指定するのに利用できるだろう。

68000のプログラムカウンタ相対アドレッシングには“ディスプレースメント付きPC相対形式”と，“インデックス付きPC相対形式”の2種類がある。それぞれ，

```
move.w 10(pc), d0
move.w 8(pc, d1), d0
```

のようにして使い，一般形は，

```
disp16(pc)
disp8(pc, index)
```

となる。基準がアドレスレジスタからプログラムカウンタに代わった以外はディスプレースメント付き/インデックス付きアドレスレジスタ間接形式と同じような格好で，ディスプレースメントの範囲なんかも同じだ。が，PC相対は多少アセンブラに優遇されていて，

```
        move.w  LABEL(pc), d0
        move.w  LABEL(pc, d1), d0
            ⋮
LABEL:  .dc.w  123
```

のようにディスプレースメントの部分にアクセスしたいアドレス自体を表すラベルを使うことが許されている。こう書いておくと，命令の置かれたアドレスとアクセスするアドレスの差をアセンブラが勝手に計算してくれるのだ[1]。もし，アセンブラが頑固で，ディスプレースメントに数字しか記述できないようだとアドレスの差をプログラマが計算しなければならないという悲惨なことになるところだ。

絶対アドレッシングの代わりにPC相対アドレッシングを使う利点としては，

1) プログラムを短くできる[2]

2) プログラムを高速にできる[3]

といったこともあるが，

3) プログラムをリロケータブルにできる[4]

というのが最大の強みであり，これこそがPC相対の

1) 細かな話になるが，68000のPC相対アドレッシングでアドレス計算の基準となるPCは，命令が置かれた先頭アドレスから2バイトだけずれている。これは68000の命令実行サイクルが，
(1) PCの位置から1ワード読み込む
(2) PCを1ワード分＝2バイト進める
(3) 命令を解釈し，実行する
という順序で行われるためだ。実効アドレスを計算するのは(3)の時点だから，PCはすでに2バイト進んでしまっているわけだ。もっとも，ディスプレースメントの計算をアセンブラにまかせてしまえば，こんなことは知らなくても困らないが。

2) 必ずしも一般的にそうだというわけではない。68000の場合はディスプレースメント付き相対形式ではディスプレースメントが16ビットに“制限されているから”32ビットの絶対アドレスで指定するより1ワード分コードが短くなるという話であって，（実際にあるかどうかは知らないが）絶対アドレスと同じだけのビット幅でディスプレースメントを指定できるようなプロセッサがあればその差はないわけだ。

3) これも68000ではの話。プロセッサによってはPC相対アドレッシングを使うとアドレス計算に時間がかかるのか，かえって遅くなる場合もある。

4) リロケータブル（relocatable：直訳すれば再配置可能）というのはメモリ上のプログラムをごっそり別のアドレスに移動しても手を加えることなく動く，つまり，どのアドレスにロードしてもそのまま動作するという意味だ。Human68kのXファイルもロードするアドレスを選ばないが，これはHuman68kがロード時にロードアドレスに応じてプログラムを書き換えている（そのための情報はXファイルが持っている）からであって，純粋な意味でのリロケータブルとは区別し，ソフトウェアリロケータブルという言葉で表現される。

存在理由でもある。ところが，残念ながら68000のPC相対アドレッシングにはちょっとした制限がある。というのも，デスティネーションオペランドには利用することができないのだ。

```
        move.w  LABEL(pc), d0
```

はできても，

```
        move.w  d0, LABEL(pc)
```

は許されない。これは68000の設計思想に関わる問題だったりする。68000は，完全ではないにしろプログラムとデータの分離という考えを打ち出していて，詳しい話はしないけどメモリ空間なんかにもそういった思想が見える。で，その思想に従い，

1) PC が指しているのはプログラムが置かれた領域であり，プログラムカウンタ相対でアクセスされるのはそのプログラムが置かれた領域である。
2) プログラムは本来実行中に書き換えるべきものではない。
3) ゆえに，PC相対で指定されるアドレスが書き換えられるはずがない。

という見事な三段論法によって，PC相対アドレッシングではメモリの参照だけができ，書き換えができないように作られている。ま，こういったお上品さが68000らしさでもあるわけだ。

が，このお陰でリロケータブルなプログラムを作ろうと思ったら，単に絶対アドレッシングをすべてPC 相対に置き換えればすむというような単純な作業では不十分で，多少の小細工が必要になってくる。具体的には，

```
        move.w  d0, work
```

を，

```
        lea.l   work(pc), a5
        move.w  d0, (a5)
```

と操作したいメモリのアドレスを PC 相対でアドレスレジスタに入れておいてアドレスレジスタ間接アドレッシングを利用する。メモリに書き込むたびにこんなことをするのは冗長なので，実際にはプログラムの先頭でアドレスレジスタを一度だけ初期化して，以降のメモリアクセスはそのアドレスレジスタを通じて行うようにすることが多い。ワークが複数ある場合にもワークエリアがひとまとめになっていれば，ディスプレースメント付きアドレスレジスタ間接形式が利用できるので，アドレスレジスタは1本しか潰さないですむ。ワークエリアが，

```
        work1:  .ds.l   1
        work2:  .ds.w   1
```

のような構成だったとすると，work2はwork1から4バイト先だから，

```
        lea.l  work1(pc), a5
```

と初期化しておけば，

```
        move.l  d0, (a5)
```

によってwork1にアクセスできるだけでなく，

```
        move.w  d1, 4(a5)
```

によってwork2へも書き込みが行えるわけだ。

Human68kの実行ファイルには，通常使われているソフトウェアリロケータブルなX形式以外に，完全リロケータブルなR形式，特定のアドレスに置かれたときにしか動作しないZ形式がある。そして，上記のような手法で絶対アドレス参照を行わずにリロケータブルに組まれたプログラムは，アセンブル・リンク後に生成されたXファイルをさらにコンバータCV.X[5]に通すことでR形式のファイルに変換することができる。

R形式のファイルはX形式のように再配置情報をファイルに含まなくてすむため[6]，若干ファイルサイズが小さくなるなどの利点がある。ただし，CV.XでR形式にコンバートするプログラムにはブロックトレースセクションとスタックセクションがあってはならない[7]という制限があって，スタック領域や大きなバッファが必要であれば，プログラム側でmallocやsetblockなどを利用してメモリを確保して使う必要がある。また，.end疑似命令で実行開始アドレスを指定[8]しても無視されるので注意が必要だ。

ディスプレースメント付きPC相対形式

実をいうとディスプレースメント付き PC 相対形式は初登場ではなく，すでに第2回から毎回必ず使っている。bra, bsr, dbra はアセンブリ言語の書式上は，

```
        bra     LABEL
        bsr     sub
        dbra.w  d0, sub
```

というように絶対アドレス形式のような書き方をするが，実際にはPC相対で分岐する命令だ。

```
        bra     LABEL(pc)
        bsr     sub(pc)
        dbra.w  d0, sub(pc)
```

のように後ろに(pc)が隠れていると思えばいい[9]。分岐先とのアドレスの差は例によってアセンブラが自動的に計算してくれている。また，bra，bsrにはディスプレースメントを8ビットに制限し命令長を1ワードに抑える形式と，ディスプレースメントが16ビットで命令長が2ワードになる形式の2種類がある[10]。

AS.Xはディスプレースメントを計算したときに8ビットの符号付き数で収まれば自動的に命令コードが短くなるように最適化してくれる。この最適化は該当するものがなくなるまで繰り返し行われるので場合によってはアセンブルに必要以上の時間がかかることもある[11]。多少プログラムが大きくなって構わないのであれば/Nスイッチを指定することでこの最適化を行わないようにもできるが，別のテとしてはソースで明確にディスプレースメントのサイズを指定する方法がある。move などの命令でサイズを指定したときのように，

```
        bra.b   LABEL
```

とか，

```
        bra.w   LABEL
```

5) CV.XはC compiler PRO-68KやTHE福袋V2.0に含まれている。

6) R形式ファイルはマシン語プログラムのメモリ上の姿そのものをファイルにしたものだ。

7) これがあるとコンバートできても正しく動作しない場合がある。

8) .end疑似命令は後ろにラベルを置くことでプログラムの途中から実行開始するような指定ができる。複数のオブジェクトをリンクしてプログラムを作成する場合なんかは，リンクの順序が乱れても問題ないよう，.endでメインルーチンの先頭アドレスを指定しておくのが望ましい。

9) ただし，実際に，
bra LABEL(pc)
というような指定をするとアセンブル時にエラーになる(世の中には許してくれるアセンブラもあるのかもしれないが……)。

10) dbraの場合は常に16ビットディスプレースメントが使用される。

11) わかりにくい表現だが，こういうことだ。分岐先のアドレスが決定されていないときには，アセンブラはとりあえず16ビットディスプレースメントと仮定してアセンブルする。その後，実際には8ビットディスプレースメントで届くということがわかれば置き換える。すると以下の命令がごっそり1ワード前にずれることになり，さっきまでは8ビットディスプレースメントでは届かなかったのが届くようになる可能性が出てくる。それを調べるためには，もう一度ソースを最初にさかのぼって見直さなければならない，というわけだ。

とか記述すればよい。特に,

```
bra.b    LABEL
```

は,

```
bra.s    LABEL
```

というように“.s(Short)”という表記をすることが許されており,たいていはこっちを使う。

braやbsrの真の姿を暴露したところで,ついでに絶対アドレスで任意のアドレスに飛べる分岐命令,サブルーチン呼び出し命令を紹介しておく。braに対応する無条件分岐命令はjmp(JuMP),bsrに対応するサブルーチン呼び出しのほうは先月顔を出したjsr(Jump to SubRoutine)だ。どちらも絶体アドレスで分岐先を指定できるだけでなく,

```
jmp     (a0)
jsr     jmptbl(pc, d0, l)
```

のように68000の豊富なアドレッシングモードを利用することができる。その代わりにjmpは無条件ジャンプ専用で,条件分岐するためにはbcsなどと組み合わせて実現するしかない。近場に条件付き相対分岐して,そこからjmpで高飛びするわけだ。

クロックと実行速度

続いては,マシン語命令の実行速度の話。これまでにも,0との比較にはcmpiを使うよりtstのほうが速いとか,ふつうのイミディエイト形式よりもクイックイミディエイト形式のほうが名前からして速いとか,いくつか具体的な例は挙げてきたが,この辺で一度まとめておこうと思う。

68000の命令の実行速度は,命令の種類,サイズ,アドレッシングモードによって変化するがだいたいの目安がある。順不同で並べてみると,

1) サイズはロングワードよりワードのほうが速いが,ワードとバイトは多くの場合同じ速度
2) 乗除算のように感覚的に動作が複雑そうな命令は遅い
3) (プロセッサ内部にある)レジスタに対するアクセスは(プロセッサ外部にある)メモリに対するアクセスより速い
4) アドレッシングモードは複雑になるにつれて遅くなる。が,単純そうな絶対ロング形式が一番遅い
5) メモリに何回もアクセスするほど遅くなる
6) オペランドを含む命令コード全体の長さが長いほど遅い
7) moveに対するclr, cmpiに対するtstのように一部の特定条件でのみ使用できる専用命令は速い(少なくとも遅くはない)
8) クイックイミディエイト形式はふつうのイミディエイト形式より速い(その代わり指定できる範囲に制限がある)

上に挙げた条件は部分的には同じことをいっている。6)の命令コードが長いほど遅いというのは,そのぶんメモリに多くアクセスするし(PCの指す位置からの命令取り込みもメモリアクセスだ),命令コードが長いということはきっとアドレッシングモードも複雑だろうし,複雑なアドレッシングモードというからにはデータレジスタ直接なんかじゃなくてメモリが対象だろうからまたメモリアクセスがあるだろうし,といくつもの条件が重なっていると考えられる。また,7)のclrやtstはオペランドがmoveやcmpiの2個に対して1個ですむぶん,8)のクイックイミディエイト形式は1ワードの命令の中にオペランドを包含しているぶん,それぞれ命令コードの全体長が短くなって~,という具合だ[12)]。

より正確な命令の実行時間を知りたければ,まともな命令表になら載っている各命令のクロックサイクル数を調べてみるとよい。命令のクロックサイクル数というのはプロセッサに与えられるクロック[13)]のパルス(⎍)何個分の時間で実行が終了するかを表し,小さければ小さいほど速い命令であることを意味している。ハードウェアに密着したプログラムでは,ある処理を行ったら何ms待たなければならない,とか,処理を何ms以内に完了しなければならない,なんてこともあり,そういう場合はクロックサイクル数にクロック周波数から計算した1クロック当たりの時間を掛けて命令の実行に必要な時間を求めたりもする[14)]。そこまでいかなくても,命令表を見ながら各命令の実行時間の差の原因を考えるの

12) clrでメモリを対象にする場合は68000は一度メモリを読み出し,それから0を書き込む。この余分なメモリアクセス1回のために,結局はmoveで0を書き込むのと同速度になってしまっている。clrがmoveより速いのは対象がデータレジスタのときに限られることになり,それすらもサイズがロングのときにはもっと速いmoveqがあるわけだ。
13) clock:コンピュータの動作基準となる信号
14) X68000のクロック周波数は10MHzだから,クロックは100万分の1秒=0.1ms=100nsとなる。

アドレスレジスタ直接形式の落とし穴

68000はデータとアドレスを明確に分離して扱っており,同じレジスタ直接アドレッシングでもデータレジスタ直接形式とアドレスレジスタ直接形式の扱いはかなり違う。第一に,バイトサイズが許されていない。

```
move.b   a0, d0
adda.b   #4, a0
```

なんかはないわけだ。これはまだいい。問題なのはデスティネーションオペランドにアドレスレジスタ直接形式,サイズにワードを指定した

```
movea.w  d0, a0
adda.w   #123, a0
```

のような場合だ。この場合はデータ転送や演算に先立ってソースオペランドは32ビットへ符号拡張され,結果のアドレスレジスタは全32ビットが影響を受けることになっている。d0.l, a0.lにそれぞれ8FFFFHが,d1.lに1が入っているときに,

```
add.w    d1, d0
adda.w   d1, a0
```

を実行すると,d0=80000H,a0=90000Hになるし,同じ条件で,

```
move.w   d1, d0
movea.w  d1, a0
```

を実行すれば,d0=80001H,a0=1Hという結果になる。この違いをはっきり理解してほしい。

ワードサイズを指定してもロングワードすべてが影響を受けるというのは感覚と一致しなくて気持ちのいいものではないが,これを逆手にとった姑息なテクニックがある。たとえば,

```
addq.l   #1, a0
subq.l   #1, a0
```

の代わりに,

```
addq.w   #1, a0
subq.w   #1, a0
```

を使うのだ。こっちのほうがちょっとだけ処理時間が短くてすむ。プログラムの実行速度にこだわるのなら,これを利用しない手はない。ただ,この方法を使うとAS.Xが警告を発してくる。アドレスレジスタに対してワードで演算しているのが不自然に見えるのだろう。わかっていて出す警告にはなんの害もないから,あんまりうるさいようだったら/Wスイッチを指定して黙らせてしまおう。

はなかなか楽しい遊び(?)だし，同じ動作をする2つの命令があったらどちらが速いか調べてみるのも有意義だろう。

表1にそうやって調べた常套句をいつくか挙げておく。表の中ほどでは，

```
addq.w   #1, a0
```

という，アドレスレジスタに対するワードサイズの演算という見慣れない形が登場しているが，これについてはコラムで触れておく。また，表の最後に示した2つのパターンでは，絶対ショートアドレス形式というマイナーなアドレッシングモードを利用している。このアドレッシングモードは16ビットで表現できる範囲の絶対アドレスを指定するもので，実効アドレスを求める際には32ビットに符号拡張される[15]。が，現バージョンのAS.X が絶対ショートアドレス形式をサポートしていないため，このままの形で使っても意味はない[16]。ここまでやるのもどうかとは思うが，どうしてもこのパターンを使いたければ，

```
PEAW    macro   data
        .dc.w   $4878, data
        endm
```

とマクロ定義しておき，

```
pea.l   10.w
```

の代わりに，

```
PEAW    10
```

のようにして使うといった工夫が必要になってくる。ここで4878Hというコードは，

```
lea.l   XXXX.w
```

のマシンコードで[17]，これを .dc.w で直接プログラムに埋め込んでいるわけだ。

マシン語の最適化

命令の実行速度の話が出たところで，最適化の一例を示そう。前回，ジャンプテーブルのもっともオーソドックスな形としてリスト1-aのようなコーディングをしてみせた。これはオーソドックスというよりは泥臭いくらいのプログラムだ。これを今月出てきたお洒落なアドレッシングモードなんかを使ってプログラムサイズ・速度の両面から重箱の隅をつつくように最適化していくとどうなるか，ちょっと考えてみたい。以下，プログラムの右側に注釈の形でつけた2つの数字は順に命令コードの長さ（単位はバイトではなくワード）と実行にかかるクロックサイクル数を表す。

まず，インデックス付きアドレスレジスタ間接形式を使えば，

```
adda.l   d0, a0             *1  8
movea.l  (a0), a0           *1  12
```

の部分は，

```
movea.l  0(a0, d0.l)        *2  18
```

とまとめられるだろう[18]。2クロックの儲けだ。ここにディスプレースメント付きPC相対形式を導入して，

```
lea.l    tbl, a0            *3  12
```

の部分を，

```
lea.l    tbl(pc), a0        *2  8
```

と書き直せば，1ワードと4クロックの儲け[19]。さらに，

```
lea.l    tbl(pc), a0        *2  8
movea.l  0(a0, d0.l), a0    *2  18
```

の2命令はインデックス付きPC相対形式で書き直すと，

```
movea.l  tbl(pc, d0.l), a0  *2  18
```

となり，まるまる2ワードと8クロックが浮く[20]。

ここまでで，プログラムの大きさが3ワード，実行時間が14クロック分短縮された。68000の全メモリ空間のどこへでも分岐できるジャンプテーブルの形としては，この辺りが最適化の限界だろう。最終的なコードをリスト1-bに示す。

もう一歩突っ込んで，(PC相対アドレッシングが有効に利用できるように）分岐できる距離をある程度制限すればリスト2に示すような形でさらに最適化を進めることもできる。リスト2-aはテーブルに分岐先アドレスを並べる代わりにbraを並べて，

```
movea.l  tbl(pc, d0.l), a0  *2  18
jsr      (a0)               *1  16
```

を，

```
jsr      tbl(pc, d0.l)      *2  22
bra.w    処理アドレス        *2  10
```

とすることで1ワードと2クロックを稼ぎ出している（braの2ワードはテーブルのぶんで相殺される)。ただ，bra.sとbra.wが混在するとテーブルとして成り立たなくなってしまうので，bra はワード分岐を明示して常に2ワードのコードになるようにしてある。もし，bra.s で届く範囲に処理ルーチンが集まっているのであれば，リスト2-bのようにすることでテーブルの大きさを半分に抑え，また，アド

表1　より高速な命令への置き換え例

0の代入	move.b #0,d0	→	clr.b d0
	move.w #0,d0	→	clr.w d0
	move.l #0,d0	→	moveq.l #0,d0
	movea.l #0,a0	→	suba.l a0,a0
0との比較	cmpi.w #0,d0	→	tst.w d0
アドレスレジスタに対する	adda.l #1,a0	→	addq.w #1,a0
定数の加減算	subq.l #8,a0	→	subq.w #8,a0
	adda.l #99,a0	→	lea.l 99(a0),a0
スタックへのプッシュ	pea.l (a0)	→	move.l a0,-(sp)
	move.l #99,-(sp)	→	pea.l 99.w
	clr.l -(sp)	→	pea.l 0.w

15) これにより，絶対ショートアドレス形式で指定可能な範囲は，
00000000H～00007FFFH
FFFF8000H～FFFFFFFFH
というメモリの最下位32Kバイトと最上位32Kバイトに制限される。なお，68000のアドレスは下位の24ビットのみが有効であり，上位8ビットは無視されるから実質的には，
00000000H～00007FFFH
00FF8000H～00FFFFFFH
となる。

16) 絶対ショートアドレス形式を指定しても絶対ロングアドレス形式と見なされてしまうため，かえってmoveを使うより遅くなってしまう。

17) このコードはデバッガ DB.XのAコマンドで1行アセンブルして調べた。デバッガではちゃんと絶対ショートアドレス形式がサポートされている。

18) この例では処理番号が8ビット以内であり，8ビット数は4倍しても10ビットに収まるから，
movea.l 0(a0, d0.w), a0
でも同じ結果が得られる。

19) tblが命令の置かれた位置から16ビットのディスプレースメントで届く距離にあることが前提になる。

20) 今度はtblが8ビットのディスプレースメントで届く距離にあることが前提になる。

リスト1

a)

```
 1:          moveq.l #0,d0
 2:          move.b  処理番号,d0
 3:          add.w   d0,d0
 4:          add.w   d0,d0
 5:          lea.l   jmptbl,a0
 6:          adda.l  d0,a0
 7:          movea.l (a0),a0
 8:          jsr     (a0)
 9:                  :
10: jmptbl:
11:          .dc.l   test0
12:          .dc.l   test1
13:          .dc.l   test2
14:                  :
```

b)

```
 1:          moveq.l #0,d0
 2:          move.b  処理番号,d0
 3:          add.w   d0,d0
 4:          add.w   d0,d0
 5:          movea.l jmptbl(pc,d0)
 6:          jsr     (a0)
 7:                  :
 8: jmptbl:
 9:          .dc.l   test0
10:          .dc.l   test1
11:          .dc.l   test2
12:                  :
```

リスト2

a)

```
 1:          moveq.l #0,d0
 2:          move.b  処理番号,d0
 3:          add.w   d0,d0
 4:          add.w   d0,d0
 5:          jsr     jmptbl(pc,d0)
 6:                  :
 7: jmptbl:
 8:          bra.w   test0
 9:          bra.w   test1
10:          bra.w   test2
11:                  :
```

b)

```
 1:          moveq.l #0,d0
 2:          move.b  処理番号,d0
 3:          add.w   d0,d0
 4:          jsr     jmptbl(pc,d0)
 5:                  :
 6: jmptbl:
 7:          bra.s   test0
 8:          bra.s   test1
 9:          bra.s   test2
10:                  :
```

c)

```
 1:          moveq.l #0,d0
 2:          move.b  処理番号,d0
 3:          add.w   d0,d0
 4:          move.w  jmptbl(pc,d0),d0
 5:          jsr     jmptbl(pc,d0)
 6:                  :
 7: jmptbl:
 8:          .dc.w   test0-jmptbl
 9:          .dc.w   test1-jmptbl
10:          .dc.w   test2-jmptbl
11:                  :
```

レス計算時の,

```
add.w     d0, d0                  *1  4
```

をひとつ省くことができるが，8ビットディスプレースメントでは遠くへは分岐できないため，実用性は薄い。

また，別のアプローチとしてはリスト2-cのような方法もある。テーブルに処理ルーチンのアドレスそのものではなく，テーブル先頭アドレスとの差がワードで並べてある。これでどうしてジャンプテーブルとして成り立つのかじっくり考えてみてもらいたい。最適化の程度としてはリスト1-bと比べ,

```
add.w     d0, d0                  *1  4
```

がひとつ減り,

```
movea.l   tbl(pc, d0.l), a0       *2  18
jsr       (a0)                    *1  16
```

が,

```
move.w    tbl(pc, d0.l), d0       *2  14
jsr       tbl(pc, d0.w)           *2  22
```

になっているので2クロックだけ速い。これに加えてテーブルの大きさが半分になっているのでメモリ効率もよい。今後ジャンプテーブルの手法を利用するときには，このリスト2-cのパターンを使うことになるだろう。

DOSコールの秘密

次の話題は，どうして,

```
DOS      _EXIT
```

と書くとDOSコールが呼び出せるのかという話。周知のことと思うが，“DOS”はDOSCALL.MACの中で,

```
DOS   macro    callno
      .dc.w    callno
      endm
```

のように定義されたマクロであり，また，“_EXIT”はやはりDOSCALL.MAC内で,

```
_EXIT    equ    $ff00
```

と定義された記号定数だ。結局,

```
DOS      _EXIT
```

はアセンブル時にアセンブラによって,

```
.dc.w    $ff00
```

に展開される。さらには“.dc.w”はオペランドに並べられたワードデータをそのままオブジェクトに吐き出すことをアセンブラに指示する疑似命令だから，この行は最終的にFF00Hというコードに変換され，オブジェクトプログラムに埋め込まれる。exit以外のDOSコールの場合も同様で，getcharならFF01H，putcharならFF02HというようにDOSコール番号そのままの形でプログラムに埋め込まれる。ここまではいいだろう。

さて，マシン語プログラムの実行は，PCを順次進めつつ，そのPCがポイントするメモリからビット列を取り出しては（それが命令だと仮定して）解釈・実行することで行われるのだった。特定のビット列はある動作と1対1に対応しており，概念的にはCPUはメモリから拾ったビット列と内部に持った命令表とを照らし合わせて該当する動作を調べることになる。

実際にはマシンコードは0000Hならこんな動作，0001Hならこんな動作と個別に決められているわけではなく，解釈しやすいように，たとえば，命令の種類を4ビット，ソースオペランドの種類を6ビット，デスティネーションオペランドを6ビットでそれぞれ表す，というようなビットフィールドになっているのがふつうだ。もちろん例外もあって，特殊な命令にはこの秩序から外れたマシンコードが割り振られていることもあるし，命令の数とマシンコードのビット長との兼ね合いから動作が割り当てられていないマシンコードの隙間も結構ある。プロセッサを

バージョンアップする過程で命令を新設するときにはこの空き部分に新命令を割り振ることになる。それでもまだ足りなければ，マシンコードの構成を2段構えにして回避したりもする。ある特定のコードを頭に置き，続くもうひとつと組み合わせて動作を定義するという，ほとんどエスケープシーケンスのノリだ。こうしてなんとかザウルスができ上がる。

それはともかく，現在のコンピュータでは，PCの指すものはとにかく命令と見なすという上述の方式が主流中の主流になっている。いわゆる暴走というヤツの大部分はなにかのはずみでPCがあらぬところを指し，偶然そこに格納されているもの（プログラムの一部かもしれないし，データかもしれないし，メモリ上に残った単なるゴミかもしれない）を強引に実行しようとして予想外の結果を引き起こしてしまった状態なわけだ。

ここで，“強引に実行しようとする”のにも2通りが考えられる。PCの指すものが命令として解釈できる場合と，できない（動作が定義されていない）場合だ。前者であれば，“PCの指すものは命令である”の掟に従い，プロセッサは平気な顔で実行を続ける。その結果，変なメモリを書き換えたりして2次被害が発生する。ただし，68000の場合は変なメモリを書き換えようとしたときにバスエラーやアドレスエラーで止まってくれる可能性が高い。また，後者の定義されていない命令を実行しようとした場合の動作はプロセッサによって（可能性としてはチップ1個1個ごとに）異なり，単に無視するかもしれないし，なにか突拍子もない動作をすることだってありうる。68000ではどうなるかというと，ちゃんとエラーを出してくれる。賢い。

DOSコールの呼び出しに使われている FFxx_H というコードに話を戻そう。結論からいうと，68000ではFFxx_Hはおろか，上位4ビットが1111Bのコード（Fxxx_H）にはいっさい命令が割り振られていない。当然，PCがこのコードを指すようなことがあれば，68000は知らない命令があるといって怒り出す。“怒る”なんていい加減な表現ではなく正しい言葉を使うと“例外処理”を始める。

68000ではバスエラーやアドレスエラー，そして動作の定義されていない命令の検出なんかをひっくるめて“例外(exception)”と呼び，なんらかの例外が発生するとその時点で実行中だった処理を中断して，ある決まった手順で例外処理ルーチンに制御を移す。例外処理ルーチンは例外ごとに用意することができる。で，X68000＋Human68kでは，バスエラー例外やアドレスエラー例外なんかのときは例のエラーメッセージを表示するルーチンが，Fxxx_Hのコードが実行されそうになったときにはDOSコールの処理ルーチンが，例外が発生するのを待ち構えているわけだ。

例外が発生したときに制御が移るアドレスはメモリの 000_H～3FF_H 番地に4バイトずつのテーブルとして用意されている。この4バイトごとのメモリを例外ベクタと呼び，例外ベクタには 00_H～FF_H のベクタ番号が振られている。バスエラー例外のベクタ番号は2，アドレスエラー例外なら3，というように例外とベクタ番号は基本的に1対1に対応している。DOSコールの呼び出し時に発生するのはベクタ番号11に割り振られた“1111系列未実装命令[21]の実行による例外”だ。ベクタアドレスは 02C_H 番地となる。

さて，未実装命令の検出によって例外が発生すると，68000 は自動的にスーパーバイザモードに移行し[22]，スタックに，

```
(sp)＝sr
2(sp)＝pc
```

となるように例外が発生した時点でのpcとsrを待避する。すでにスーパーバイザモードになっているから参照されるのはsspだ。その後，ベクタ番号11の内容がpcに取り込まれ，DOSコールの処理が始まる。

ここでは，まず，DOSコール番号を調べなければならない。未実装命令の実行による例外時には，スタックに待避されたpcは例外を引き起こした命令コード自身を指すことになっているから，たとえば，

```
movea.l  2(sp), a0
move.w   (a0)+, d0
move.l   a0, 2(sp)
```

によってd0にDOSコール番号が取り出される。a0はポストインクリメントしてスタックに戻す。この時点でスタック上には，

```
(sp)＝DOSコール呼び出し前のsr
2(sp)＝戻りアドレス
```

が保存された形となる。あとでユーザープログラムに戻るときには，スタックからsrとpcを同時に取り出すrte(Return from Exception) という命令[23]を

21) 上位4ビットが1111Bだからこう呼ぶ。同様に 68000では上位4ビット1010Bのコード（Axxx_H）も未実装になっており，こちらにはベクタ番号10が割り当てられている。1010系列はHuman68kではずっと空きになっていたが，最近，SX-WindowのTool Box の呼び出し用に使われるようになった。

22) srの第13ビットが1ならスーパーバイザモード，0ならユーザーモードを表しており，このビットを操作することでスーパーバイザモード↔ユーザーモードの切り換えが行える。ただし，srの上位バイト（第7～15ビット）はスーパーバイザモードでなければ書き換えることができない。ユーザーモードからスーパーバイザモードへの切り換えは“なんらかの例外を発生させる”ことで行う。

23) rteは“特権命令”であり，スーパーバイザモードでなければ実行できない。ユーザーモードで使うとエラー（特権違反による例外）が発生する。

サンプルプログラムを動かすうえで

この連載ではよく使う定数類の定義はインクルードファイルにまとめておいてソースに取り込んだり，以前作ったサブルーチンをライブラリとして使ったりといった使い回しをよくやっている。インクルードファイルとしてはDOSコール番号を定義した“DOSCALL.MAC”とコントロールコードなどの定数を定義した“CONST.H(1989年10月号掲載)”は毎回必ず使う。

DOSCALL.MACは本来はXCがコンパイル時に利用するファイルであり，THE福袋V2.0のユーザーにはDOSコール一覧を元に自力で作成してもらうしかなかったわけだが，今月号の付録ディスクに収録されたからいまから始めようという人はこれを使えばよい。また，付録ディスクにはIOCSコール番号を定義した“IOCSCALL.MAC”も収められている。これは来月以降使うことになるだろう。あと，“DRIVER.H(1990年4月号掲載)”もいつかまた利用することがあるかもしれない。

サブルーチンとしては，

- PUTDEC
- FPUTDEC
- UTOA　（以上1989年12月号）
- ITOH
- MEMOFF
- CHILD
- COMMAND（以上1990年1月号）

は今後も頻繁に利用すると思われる。AR.Xがあれば，ライブラリファイルにしておくのが便利だろう。

24) F0xxHやF1xxHなんかのDOSコールに関係のない未実装命令をはじく必要がある。
25) XCには"FEFUNC. H" というファイルが含まれており，この中でFLOATn.Xの内部ファンクション番号が定義されている。

使い，srを復帰しつつ（これによりユーザーモードに移行することができる），DOS コール呼び出し直後から処理を再開する。

DOSコール番号を取り出したら，上位バイトが間違いなくFFHかどうかを確認する[24]。違ったらエラー処理に即飛んでしまう。上位バイトがFFHであることが確認されたら，スタックに積まれているはずのパラメータの先頭アドレスをa6に入れたうえで，下位バイトに応じてジャンプテーブルの手法を使って各DOSコールの処理ルーチンへ分岐する。実際にはもう少し複雑なのだが，だいたいはこんな感じだ。

なお，FLOATn.Xによってサポートされる演算ファンクションコールはDOSコールと同じ1111系列の未実装命令であるFExxHを使って呼び出される[25]。FLOATn.Xは組み込み時に例外ベクタ番号11の内容がFLOATn.Xの内部を指すように変更し，その状態で1111系列の未実装命令が実行されると，まず，FLOATn.XによってFExxHかどうかがチェックされ，そうでなければ元のDOSコールの処理ルーチンに制御が移される。

ここで関連DOSコールを紹介しておく。

・DOSコール$FF35 intvcg

```
move.w  例外ベクタ番号，－(sp)
DOS     _INTVCG
addq.l  2, sp
```

例外ベクタの内容を読み出すDOSコールだ。例外ベクタ番号を指定すると，そのベクタ内容がd0.lに返ってくる。例外ベクタ番号の代わりにDOSコール番号を指定することもでき，その場合は該当するDOSコールの処理ルーチン先頭アドレス（これもベクタと呼ぶことがある）が返ってくる。

・DOSコール$FF25 intvcs

```
move.l  セットするベクタ内容，－(sp)
move.w  例外ベクタ番号，－(sp)
DOS     _INTVCS
addq.l  6, sp
```

intvcgの逆で，指定した例外ベクタの内容を変更する。リターン時のd0.lには変更する前のベクタ内容が返ってくる。やはりDOSコール番号を指定することができ，これによって個々のDOSコールの処理ルーチンを自作のルーチンと差し換えたりすることができる。

今月のプログラムASX.X

以上が，入門編で書ききれなかった“こまごまとした話”の（だいたい）全部だ。結構強引に詰め込んであるから拒否反応を起こした人もいるかもしれない。確かに多少の積み重ねを要求してはいるが，それだけのものはこの1年間で養ってもらえたと思っている。では，最後に今月の話のまとめとして実用プログラムを1本。AS.X に機能を追加する補助プログラムとでもいおうか。AS.X が出力するエラーメッセージを横取りして，ED.X のタグファイル形式のエラーファイルを出力するプログラム，ASX.Xだ（リスト3）。

ASX［スイッチ］ソースファイル名

のようにして使用すると，アセンブルしながらカレントディレクトリにAS.ERRの名前でエラーファイルを作成する。

実際にアセンブルする部分はチャイルドプロセスで AS.X を起動することで行っているので，プログラム自体は（アセンブラを作ることを思えば）ごく短いものになっている。リスト4のテスト用プログラム（2カ所に誤りがある）をアセンブルしたときの実行結果を図1に示す。図1-aがAS.Xの出すエラーメッセージで，図1-bが同時にASX.Xによって作成されるエラーファイルだ。このエラーファイルをED.Xで読み込み，タグジャンプを利用すれば効率的にエラーを修正できる。ちなみに，ED.X のタグジャンプ機能とは，テキストの任意の行の頭から，

ファイル名　　行番号

と書いておき，この行にカーソルを合わせてESC・Vを押すと自動的にそのファイルを読み込んで指定行にカーソルを移動してくれるというものだ。

AS.Xのメッセージを盗み取るには，AS.Xがメッセージ表示に利用しているDOSコールをintvcsで自作のルーチンに置き換えればよい。調べてみたところ AS.X はDOSコールwriteを使って標準出力にメッセージを書き出していることがわかったので，ここを乗っ取る。理屈はこれでよいのだが，現実にはいくつかの障害があった。

1)　writeの呼び出しの中からメッセージ出力だけを抜き出す方法

AS.X はオブジェクトファイルを作成するプログラムなのだから，writeの呼び出しが必ずしもメッセージの出力とは限らない。メッセージ以外は本来のwrite の処理ルーチンに飛ばしてやる必要がある。本来の処理ルーチンはintvcsでベクタを書き換えたときの戻り値として得られ，また，メッセージの出力

図1

a) AS.Xのエラーメッセージ

```
X68k Assembler v1.01 Copyright 1987 SHARP/Hudson
line    4 undefined symbol error
line    5 illegal size error
undefined symbol
_PUTCHR
   2 Fatal error(s)
```

b) ASX.Xが作成するタグ付きエラーファイル

```
X68k Assembler v1.01 Copyright 1987 SHARP/Hudson
A:¥WORK¥TEST.S      4 undefined symbol error
A:¥WORK¥TEST.S      5 illegal size error
undefined symbol
_PUTCHR
   2 Fatal error(s)
```

かどうかは出力先のファイルハンドルが標準出力かどうかで判断できるだろう。

2) メッセージからエラーメッセージを抜き出す

AS.Xのエラーメッセージは必ず“line”＋1個以上のスペース＋エラー行で始まるから，行先頭の数文字を比較して一致するようならエラーメッセージと判断する。ASX.Xではついでに警告メッセージも同様の方法で抜き取っている。

3) 安全性への配慮

乗っ取ったDOSコールはプログラム終了時に元に戻しておかなければならない。これは保存しておいた元のベクタ内容を終了前に再びintvcsでセットし直すことで解決する。が，これだけでは，BREAKキーなどで中断されたりするとベクタを元に戻さないうちに親プロセスに帰ってしまう可能性があり，不十分だ。そこで“プロセスを中断した際の戻りアドレス”を指定して，ダイレクトに親に帰らないよう細工する。これにもやはりintvcsを使う。

intvcsでDOSコール番号としてFFF1Hを指定すると“エラーが発生し中断した際の戻りアドレス”，FF

リスト4 TEST.S

```
1:          .include        doscall.mac
2:
3:          move.w  #'0',-(sp)
4:          DOS     _PUTCHR
5:          adda.b  #2,sp
6:          DOS     _EXIT
```

F2Hを指定すると“BREAKキーが押されてブレイクチェックに引っ掛かったときの戻りアドレス”をそれぞれ設定することができる。リスト3では24～29行がこの処理で，エラーやBREAKキーで中断されたときには301行に飛んでくるように設定している。301行以下ではwriteのベクタを元に戻してから中断された旨のメッセージを出力して，改めてエラー終了している。リスト3では“ベクタを書き換える前に中断された場合”に備え，ベクタを書き換えずみかどうかのフラグを設けて，万全を期している。

ポイントはこんなものだ。細部については注釈をもとに読んでもらえることだろう，きっと。

次回はIOCSコールをちょこちょこと使って，マウスと（今後はメインになるはずの）グラフィックのほんのさわりをやる予定でいる。

リスト3 ASX.S

```
  1: *       AS.Xのエラーメッセージを横取りして
  2: *               エラーファイルを作成する
  3: *
  4:         .include        doscall.mac
  5:         .include        const.h
  6:         .include        files.h
  7: *
  8:         .xref   memoff
  9:         .xref   child
 10: *
 11: STRMAX  equ     256                     *文字列の最大長
 12: BUFSIZ  equ     300                     *1行バッファの大きさ
 13: *
 14:         .text
 15:         .even
 16: *
 17: ent:
 18:         lea.l   mysp(pc),sp             *spの初期化
 19:
 20:         bsr     memoff                  *余分なメモリを開放する
 21:
 22:         bsr     chkarg                  *コマンドラインの解析
 23:
 24:         pea.l   break(pc)               *中断時の戻りアドレスを
 25:         move.w  #_CTRLVC,-(sp)          *  セット
 26:         DOS     _INTVCS                 *
 27:         move.w  #_ERRJVC,(sp)           *
 28:         DOS     _INTVCS                 *
 29:         addq.w  #6,sp                   *
 30:
 31:         bsr     do                      *メイン処理
 32:
 33:         DOS     _WAIT                   *AS.Xの終了コードを得て
 34:         move.w  d0,-(sp)                *  それをそのまま持って
 35:         DOS     _EXIT2                  *  終了する
 36:
 37: *
 38: *       メイン処理
 39: *
 40: do:
 41:         bsr     set_vector              *DOSコールのベクタを書き換える
 42:
 43:         lea.l   cmdlin(pc),a0           *子プロセスを起動するために
 44:         lea.l   prog(pc),a1             *  コマンドラインを作成する
 45:         bsr     strcpy                  *
 46:         movea.l a2,a1                   *
 47:         bsr     strcpy                  *
 48:
 49:         pea.l   cmdlin(pc)              *子プロセス起動
 50:         bsr     child                   *
 51:         addq.w  #4,sp                   *
 52:         tst.l   d0                      *
 53:         bmi     error1                  *負ならエラー
 54:
 55:         bsr     reset_vector            *ベクタを元に戻す
 56:
 57:         rts
 58:
 59: *
 60: *       DOSコールwriteが発動されるとここに来る
 61: *
 62:         .offset 0
 63: *
 64: FNO:    .ds.w   1               *ファイルハンドル
 65: DATPTR: .ds.l   1               *出力データへのポインタ
 66: DATLEN: .ds.l   1               *出力データのバイト数
 67: *
 68:         .text
 69: *
 70: write:
 71:         cmpi.w  #STDOUT,FNO(a6) *出力先は標準出力か?
 72:         bne     write0          *そうでなければ
 73:                                 *  オリジナルの処理ルーチンへ
 74:
 75:         movem.l d1-d2/a0-a3,-(sp)       *{
 76:
 77:         movea.l bufptr(pc),a0   *a0=バッファ内の次の位置
 78:         movea.l DATPTR(a6),a1   *a1=出力データ
 79:         move.l  DATLEN(a6),d1   *d1=出力データのバイト数
 80:         lea.l   buff+BUFSIZ-1(pc),a2
 81:                                 *a2=バッファの上限
 82:                                 *（終端コードの分を考慮）
 83: wrt0:   move.b  (a1)+,d0        *1バイト取り出し
 84:         move.b  d0,(a0)+        *  バッファに転送する
 85:         cmpi.b  #LF,d0          *1行分溜まったか?
 86:         beq     wrt1            *  そうなら1行出力
 87:         cmpa.l  a2,a0           *バッファの上限に達したか
 88:         bcs     wrt2            *  そうでないならスキップ
 89: wrt1:   bsr     putline         *1行出力する
 90: wrt2:   subq.l  #1,d1           *データのカウンタを1減らし
 91:         bne     wrt0            *  <>0なら繰り返す
 92:
 93:         move.l  a0,bufptr       *ポインタ更新
 94:
 95:         movem.l (sp)+,d1-d2/a0-a3       *}
 96:         *
 97: write0: jmp     0               *オリジナルのwriteへ
 98: write_org       equ     write0+2
 99:
100: *
101: *       1行分の処理
102: *
103: putline:
104:         move.w  efno(pc),d2     *d2=エラーファイルのファイルハンドル
105:         bne     putln0          *d2<>0ならファイルはオープン済み
106:
107:         move.w  #ARCHIVE,-(sp)  *エラーファイルを新規作成
108:         pea.l   efname(pc)      *
109:         DOS     _CREATE         *
110:         addq.w  #6,sp           *
111:         move.w  d0,d2           *
112:         bmi     putln3          *エラーでなければ
113:         move.w  d2,efno         *  ファイルハンドルをワークへ
114:
115: putln0: clr.b   (a0)            *文字列の終端コードを書き込む
116:         lea.l   buff(pc),a0     *a0=出力するデータ先頭
117:
118:         lea.l   errstr(pc),a3   *エラーメッセージか?
119:         bsr     strcmp          *
120:         beq     putln1          *  そうなら細工してから出力
121:
122:         lea.l   wrnstr(pc),a3   *警告メッセージか?
123:         bsr     strcmp          *
124:         bne     putln2          *  そうでなければ生のまま出力
125:
126:                         *エラー/警告メッセージだった場合
127: putln1: move.w  d2,-(sp)        *ソースファイル名を
128:         pea.l   fname(pc)       *  エラーファイルへ書き出す
129:         DOS     _FPUTS          *
130:         addq.w  #6,sp           *
131:                 *a0はエラーメッセージ中の行番号を指している
132:
133: putln2: move.w  d2,-(sp)        *残りのメッセージを
134:         move.l  a0,-(sp)        *  エラーファイルへ書き出す
```

```
135:         DOS     _FPUTS          *
136:         addq.w  #6,sp           *
137:
138: putln3: lea.l   buff(pc),a0     *ポインタ初期化
139:         rts
140:
141: *
142: *       a0の指す文字列先頭がa3の指す文字列と
143: *                 一致するかどうか調べる
144: *       一致した場合     Z=1,a0=一致した部分の直後
145: *       不一致の場合     Z=0,a0は保存される
146: *
147: strcmp:
148:         move.l  a0,-(sp)        *a0=被比較文字列先頭
149:                                 *a3=比較部分文字列先頭
150: strcp0: tst.b   (a3)            *比較文字列がもうなければ
151:         beq     strcp1          *  一致した
152:                                 *そうでなければ
153:         cmpm.b  (a0)+,(a3)+     *  比較してみて
154:         beq     strcp0          *  一致している間は繰り返す
155:                                 *不一致が検出されたら
156:         movea.l (sp)+,a0        *被比較文字列を復帰して
157:         rts                     *  戻る
158: strcp1: addq.w  #4,sp           *待避してあったa0はもういらない
159:         rts                     *一致した
160:
161: *
162: *       DOS $ff40 writeのベクタを書き換える
163: *
164: set_vector:
165:         pea.l   write(pc)       *置き換える処理ルーチン先頭
166:         move.w  #_WRITE,-(sp)   *
167:         DOS     _INTVCS         *
168:         move.l  d0,write_org    *元のベクタを待避
169:         st.b    hooked          *ベクタを書き換えた印を立てる
170:         addq.w  #6,sp
171:         rts
172:
173: *
174: *       DOS $ff40 writeのベクタを元に戻す
175: *
176: reset_vector:
177:         tst.b   hooked          *ベクタを書き換えていないなら
178:         beq     rvec0           *  何もしない
179:
180:         move.l  write_org(pc),-(sp)     *WRITEの元のアドレス
181:         move.w  #_WRITE,-(sp)           *
182:         DOS     _INTVCS                 *
183:         addq.w  #6,sp                   *
184:
185:         clr.b   hooked          *フラグをクリア
186:
187: rvec0:  rts
188:
189: *
190: *       コマンドラインの解析
191: *
192: chkarg:
193:         addq.w  #1,a2           *a2=コマンド行文字列先頭
194:         move.l  a2,-(sp)        *あとで使うから保存しておく
195:
196:         bsr     fstarg          *空白とスイッチをスキップする
197:         lea.l   temp(pc),a0     *最初のスイッチではない単語を
198:         bsr     getarg          *  ソースファイル名と見なし
199:                                 *  temp以下に取り出す
200:
201:         movea.l (sp)+,a2        *a2=AS.Xへ渡すパラメータ
202:
203: *fname以下にフルパスのソースファイル名+TABの文字列を作成する
204:         lea.l   fname(pc),a0    *a0=ファイル名格納領域
205:         tst.b   (a2)            *パラメータがもともとなければ
206:         beq     ckarg1          *   すぐ抜ける
207:         pea.l   nambuf(pc)      *ファイル名を展開してみる
208:         pea.l   temp(pc)        *
209:         DOS     _NAMECK         *
210:         addq.w  #8,sp           *
211:         tst.l   d0              *
212:         bne     error2          *負ならエラー
213:         lea.l   nambuf(pc),a1   *
214:         bsr     strcpy          *ドライブ名+パスに
215:         lea.l   nambuf+NAME(pc),a1
216:         bsr     strcpy          *  ファイル名を加える
217:         lea.l   nambuf+EXT(pc),a1
218:         tst.b   (a1)            *拡張子は省略されているか?
219:         bne     ckarg0          *あればよし
220:         lea.l   sext(pc),a1     *なければ'.S'を補う
221: ckarg0: bsr     strcpy          *拡張子を加える
222:
223:         move.b  #TAB,(a0)+      *ついでにTABを付け加えておく
224:
225: ckarg1: clr.b   (a0)            *終端コード
226:
227:         rts
228:
229: *
230: *       コマンド行中の
231: *         スイッチではない最初の単語位置を得る(a2)
232: *
233: fstarg:
234:         bsr     skipsp          *スペースをスキップ
235:
236:         cmpi.b  #'/',(a2)       *パラメータの先頭が
237:         beq     farg0           *  /,-であれば
238:         cmpi.b  #'-',(a2)       *
239:         bne     farg1           *
240: farg0:  bsr     skipsw          *スイッチ1個をスキップ
241:         bra     fstarg          *スイッチがなくなるまで繰り返す
242:
243: farg1:  rts
244:
245: *
246: *       スイッチ1個をスキップする
247: *
248: skipsw:
249:         addq.w  #1,a2           *'/'や'-'の分を進めて
250:         lea.l   temp(pc),a0     *  temp以下に
251: *       bra     getarg          *  1語転送する
252:                                 *(転送した文字列は使わない)
253:
254: *
255: *       a2の指す位置からパラメータ1つ分をa0の指す領域へコピーする
256: *
257: getarg:
258:         move.l  a0,-(sp)        *{レジスタ待避
259: gtarg0: tst.b   (a2)            *1)文字列の終端コードか
260:         beq     gtarg1          *
261:         cmpi.b  #SPACE,(a2)     *2)スペースか
262:         beq     gtarg1          *
263:         cmpi.b  #TAB,(a2)       *3)タブか
264:         beq     gtarg1          *
265:         cmpi.b  #'-',(a2)       *4)ハイフンか
266:         beq     gtarg1          *
267:         cmpi.b  #'/',(a2)       *5)スラッシュ
268:         beq     gtarg1          *
269:         move.b  (a2)+,(a0)+     *  が現れるまで転送を
270:         bra     gtarg0          *  繰り返す
271: gtarg1: clr.b   (a0)            *文字列終端コードを書き込む
272:         movea.l (sp)+,a0        *}レジスタ復帰
273:         rts
274:
275: *
276: *       コマンド行先頭のスペースをスキップする
277: *
278: skpsp0: addq.w  #1,a2           *ポインタを進め
279:                                 *繰り返す
280: skipsp:                         *サブルーチンはここから始まる
281:         cmpi.b  #SPACE,(a2)     *スペースか?
282:         beq     skpsp0          *  そうなら飛ばす
283:         cmpi.b  #TAB,(a2)       *TABか?
284:         beq     skpsp0          *  そうなら飛ばす
285:         rts
286:
287: *
288: *       文字列の複写
289: *       リターン時a0は文字列末の00Hを指す
290: *
291: strcpy:
292:         move.b  (a1)+,(a0)+     *1文字ずつ
293:         bne     strcpy          *終了コードまでを転送する
294:         subq.w  #1,a0           *a0は進み過ぎている
295:                                 *a0は文字列末の00Hを指す
296:         rts
297:
298: *
299: *       中断/エラー終了
300: *
301: *
302: break:  lea.l   brkmes(pc),a0   *^Cなどによる中断
303:         bra     errout
304: error1: lea.l   errms1(pc),a0   *AS.Xが起動できない
305:         bra     errout
306: error2: lea.l   errms2(pc),a0   *不正なファイル名
307:         *
308: errout: bsr     reset_vector    *ベクタを元に戻す
309:
310:         move.w  #STDERR,-(sp)   *標準エラー出力へ
311:         move.l  a0,-(sp)        *  メッセージを
312:         DOS     _FPUTS          *  出力する
313:         addq.w  #6,sp           *
314:
315:         move.w  #1,-(sp)        *終了コード1を持って
316:         DOS     _EXIT2          *  エラー終了
317:
318: *
319: *       データ&ワーク
320: *
321:         .data
322:         .even
323: *
324: bufptr: .dc.l   buff            *バッファ内の次の書き込み位置
325: efno:   .dc.w   0               *エラーファイルのファイルハンドル
326:                                 *=0ならばオープンされていない
327: hooked: .dc.b   0               *ベクタ書き換え済みかどうかのフラグ
328: prog:   .dc.b   'AS.X ',0       *子プロセスとして起動するプログラム名
329: *
330: errms1: .dc.b   'ASX:AS.Xが起動できませんでした',CR,LF,0
331: errms2: .dc.b   'ASX:ソースファイル名の指定に誤りがあります',CR,LF,0
332: brkmes: .dc.b   'ASX:中断しました',CR,LF,0
333: crlfms: .dc.b   CR,LF,0
334: *
335: errstr: .dc.b   'line ',0               *AS.Xのエラーメッセージ冒頭
336: wrnstr: .dc.b   'Warning: Line ',0      *AS.Xの警告メッセージ冒頭
337: efname: .dc.b   'AS.ERR',0              *エラーファイル名
338: sext:   .dc.b   '.S',0                  *ソースファイルの拡張子
339: *
340:         .bss
341:         .even
342: *
343: cmdlin: .ds.b   STRMAX          *コマンドライン作成用
344: fname:  .ds.b   STRMAX          *アセンブルするファイル名(フルパス)
345: temp:                           *コマンドラインパラメータ1個分のバッファ
346:                                 *  (ダミー)
347: buff:   .ds.b   BUFSIZ          *AS.Xのメッセージを1行溜め込むバッファ
348: nambuf: .ds.b   NAMBUFSIZ       *nameck用バッファ
349: *
350:         .stack
351:         .even
352: *
353: mystack:
354:         .ds.l   1024            *スタック領域
355: mysp:
356:         .end
```

終わりだからターミナルなのよ

Iwai Ippei
満開製作所 祝 一平

C調言語講座も今回でオ・シ・マ・イ。というわけでターミナルソフトで「終端の美」を飾ることにしましょう。パソコン通信は一般的ですが「使えるソフト」はまだまだ少ない現状。有終のプログラムは基礎の基礎ですが，これを発展させるのはあなた自身です。いいもの作ってネットにアクセスしてみましょう。

「終わりだからターミナルなのよ（よっ，座布団1枚）」，とゆーわけで，今回で最終回である（パチパチパチ）。んで，何をしようかと思ったのであるが，ここらでちょいとターミナルソフトでも作ってみようかと思うわけである。

タコなターミナルソフト

最近では賢いターミナルソフト（その代わり特定のネット用で汎用性がなかったりする）も一部で出現しつつあるようだが，基本的に現在のものはたいていがスットコドッコイである。たとえば，ネットがすいている時間帯（平日の昼間とか）にオートログインして，自分宛のメールや目的のファイルをダウンロードして抜けてくる，などという機能を備えたものは聞かない（探せばあるのかもしれないけど）。また，あらかじめ各ネット用のヘルプファイルを用意しておいて，必要なときにはさっと呼び出せる機能とかもあったほうがいいではないか（いちいちネットのヘルプコマンドを使うと遅くてどうしようもない）。まあ，それ以外にも，ホスト側にも頑張ってほしい問題点は山のようにあるのだが，ここでは触れないでおく。

とにかく現在のような「文字が2400bpsのと～ろとろ，コマンドはキーボードから打ち込む1～3文字のアルファベット」なんてのは実にだるくていけない。言ってみれば現在のネット環境はひと昔前の「テキストアドベンチャー」なのである。なにも「ダンジョンマスター」とまでは言わないが，せめてドラクエぐらいにはなってほしいもんだ。などと言いながら，それほどではない通信ソフトをデッチ上げていくことにしよう。

ターミナルソフトの条件

まず，ターミナルソフトが最低限行わなければならないことは次の2つである。

1) キーボードから入力があったらRS-232Cに送り出す
2) RS-232Cから入力があったら画面に表示する

たとえばキーボードの「A」キーが押されたとしても，それはあくまでRS-232Cに「垂れ流し」するだけであって，決して画面に表示してはいけないのだ。しかし，やったことのある人は知っているであろうが，通信をしているときは，自分が打った文字が画面に出てくるのである（たとえば「X」キーを押すと，画面に「X」が表示される）。なぜそうなるかというと，それは「ホストがエコーバックしているから」なのである（モデムへのコマンドなどに関しては，モデムが横取りして勝手にエコーバックしていることがある。たとえば「ATDT」など）。よーするに，自分の入力した文字を電話回線上で1回往復させてから自分のマシンの画面に表示させているわけである。

で，なんでそんなことになっているのかというと，それは当然NTTの陰謀である……というのはウソ度80％である。これは，ターミナルがもともとは大型コンピュータにつながる，あまり賢くない入出力装置だったことに由来するらしい。ま，結局は因襲だな。よく考えれば，実はパソコンでも，同じようなことになっている。つまり，キーが押されたのを検知して，CPUが「むむ，Aのキーが押されたな。それではCRTにAと表示してあげましょう」なんてことをしているわけで，んでもって，そして通信の場合この「CPU」に相当するのが電話回線の向こうにいる「ホストコンピュータ」なのである。

繰り返すが，パソコン通信では「ホストがいちいちエコーバックしている」のである。だから，パソコン同士をクロスケーブルで接続して，お互いにターミナルソフト（ホストプログラムではない）を起動して通信しようとすると，自分の打った文字が確認できなくて，なかなかヤミナベである。

これが全二重の通信である（これ以外に半二重もあるわけだが電話を使った有線通信では無縁なので触れないでおく）。

で，ターミナルソフトにはあると便利な機能が2つある。それは，

1) ログファイル（通信の記録）を取る機能
2) ダウンロード，アップロード機能

ログファイルであるが，これは電話代の節約のため必要な部分をあとからゆっくり読むのに欠かせないものである。また，ダウンロード，アップロード機能は，ホストとファイルのやりとりをするために便利であることは言うまでもないが，実はこいつにはちょいと問題がある。それは，普通の通信では機械語プログラムが送れないということである（まあ，常識だが)。基本的には，改行などの一部を除いて，ASCIIコードが20H未満のコードは送らないのである。そこで，機械語プログラムをやりとりするための方法がXMODEMである（最近はより進化し高機能になったYMODEMやZMODEM，さらにその他などもある)。これはプロトコル（やりとりのための決めごと）の一種である。で，XMODEMにもいろいろなレベルのものがあるのだが，今回は一番シンプルなものを作ってみた。X68000用としては，XMODEM機能をもったプログラムとしてFXM.Xなどがネットで出回っているそうなので，そちらも参考にしていただきたい。

プログラムの説明

簡単ターミナルソフト「TTerm.C」(リスト1) のほうは別に問題はないだろう。あるとすれば91～102行のgetcon( )である。ここでは，キーボードからの「CR」(=0DH) の入力を0DH，0AHの2バイトに変換するために，ちょっと複雑なことをしている。なんでこんなことをしなければいけないかというと，それはビル・ゲイツがヒルアンドンだからである。というわけで，どーして改行がいちいち2バイトなんだよ。メモリの無駄ぢゃないか，と文句を言っておく。それと，もうひとつ注意が必要なのが，プログラム中ではキーボードからctrl-Cなどが入力される可能性があるので，putchar( )やprintf( )などのヤワな入出力関数は使えないということである（ところどころで使っているけど)。よって，低水準の入出力関数が多くなっている。

使い方であるが，[F10]が終了，[SHIFT]＋[F9]がダウンロード，[SHIFT]＋[F10]がアップロードなどなどである。ログファイルは，プログラムが勝手に名前をつけることになっている。

で，このプログラムはあくまで骨格であり，改造しやすさに重点をおいて作ったのであるから，チマチマと自分用の機能をつけ加えていかないかぎり決して便利なものではない。その点は，心得ておくように。

次にXM0.C（リスト2）である。これはXMODEMをサポートしたプログラムであるが，XMODEMの中でももっとも単純な「1ブロック＝128バイト，エラー検出はチェックサム」という，始祖鳥なものである。現在はもっと「ええもん」があるのだが，ま，あくまで原型ということである。図1がそのプロトコル，図2がブロックの形式である。

このプログラムを作っていて，ボケたのが，Xon/off制御が使えないということである（だってバイナリデータをやりとりするんだもん)。そこで，270，271行でXon/off制御の解除，279行で元に戻すということをやっている。

使い方は，

XM0 /D ファイル名　で，ダウンロード
XM0 /U ファイル名　で，アップロード

である。なお，ダウンロードのときはファイル名を省略すると，適当な名前をつけるようにしてある。また，どうやらXMODEMというのは先に送信側を起動すべきものらしいので，気をつけていただきたい。

やっぱりこの手のものは使い込んでみないとわからない部分があるので少々不安である。なお，今月のプログラムを制作，デバッグするにあたり，頼藤，桒野両氏の御協力を得た。どーもありがとさん。では，また逢う日まで。

図1　XMODEMでのやりとり

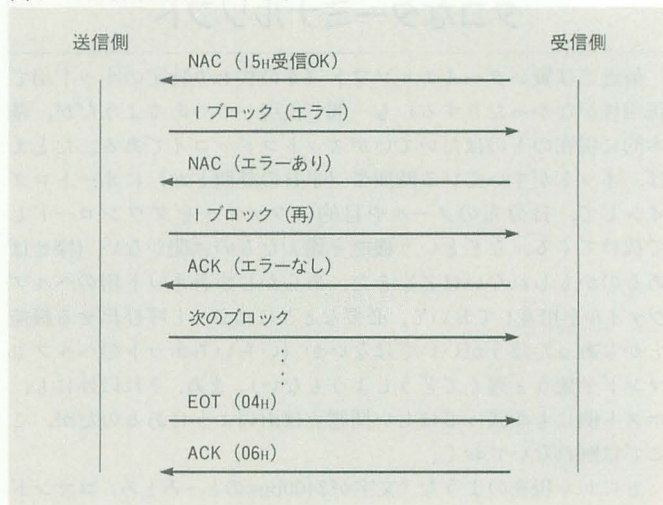
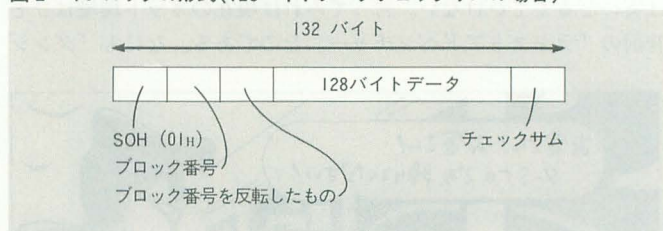



図2　1ブロックの形式（128バイトデータチェックサムの場合）

リスト1　簡単ターミナルソフト(TTerm.C)

```
 1: /* 簡単ターミナルソフト : TTerm.c*/
 2:
 3: #include    <doslib.h>
 4: #include    <iocslib.h>
 5: #include    <stdio.h>
 6:
 7: #define   CR   0x0d
 8: #define   LF   0x0a
 9:
10: UBYTE fkeybuf0[712];
11: FILE *logfp;
12:
13: void
14: main()
15: {
16:     void init(), finis();
17:     void putrs(), putcon(), ulfile(), dlfile();
18:     int f;
19:     UBYTE c;
20:
21:     init();
22:
23:     while(1) {
24:         if (f = check_func()){/*ファンクションキーが押されたか？*/
25:             switch(f) {
26:             case 10:   /* F10 */
27:                 finis();
28:             case 19:    /* SHIFT-F01 */
29:                 ulfile();
30:                 break;
31:             case 20:    /* SHIFT-F02 */
32:                 dlfile();
33:                 break;
34:             default:
35:                 break;
36:             }
37:         }
38:
```

```
39:         if (isense()) {      /* RS-232Cのバッファサイズ */
40:             c = getrs();   /* あるなら取り出す */
41:             putcon(c);   /* 画面に表示する */
42:         }
43: 
44:         if (osense()) {      /* 送信可能か? */
45:             if (c = getcon()) {   /* キーボードから読み出す */
46:                 putrs(c);   /* あるならRS-232Cに送る */
47:             }
48:         }
49:     }
50: }
51: 
52: int
53: xmget()
54: {
55:     return(0);   /* 正常終了 */
56: }
57: 
58: int
59: xmput()
60: {
61:     return(0);   /* 正常終了 */
62: }
63: 
64: /* 入力可能か */
65: int
66: isense()
67: {
68:     return(LOF232C());   /* RS-232Cのバッファサイズ */
69: }
70: 
71: /* 出力可能か */
72: int
73: osense()
74: {
75:     return(OSNS232C());   /* 送信可能か? */
76: }
77: 
78: int
79: getrs()
80: {
81:     return(INP232C());   /* 取り出す */
82: }
83: 
84: void
85: putrs(c)
86: int c;
87: {
88:     OUT232C(c);   /* あるならRS-232Cに送る */
89: }
90: 
91: int key1 = 0;
92: 
93: int
94: getcon()
95: {
96:     if (key1 == CR) {   /* トリック */
97:         key1 = 0;   /* CR の入力を CR+LF にする */
98:         return(LF);
99:     } else {
100:        return(key1 = INPOUT(0xff));/*キーボードから読み出す*/
101:    }
102: }
103: 
104: int ccc = 0;
105: 
106: void
107: putcon(c)
108: int c;
109: {
110:     INPOUT(c);   /* 画面に表示する */
111:     if (logfp) {
112:         fputc(c,logfp);
113:     }
114: }
115: 
116: /* ファンクションキーをチェックする */
117: /* SHITF -> +10
118:    CTRL  -> +20
119:    OPT.1 -> +30
120:    OPT.2 -> +40
121: */
122: int
123: check_func()
124: {
125:     int data,kc;
126: 
127:     if (data = (BITSNS(0xc) & 0b11111000)) {   /* F1-5 */
128:         kc = log(data)-2;
129:     } else if (data = (BITSNS(0xd) & 0b00011111)) {/*F6-10*/
130:         kc = log(data)+6;
131:     } else return(0);
132: 
133:     if (data = (BITSNS(0xe) & 0b00001111)) {
134:         kc += (log(data)+1)*10;
135:     }
136:     return(kc);
137: }
138: 
139: UBYTE *fkeyname[] = {
140: "はろ","どもども","¥xfe","¥xfe","¥xfe",
141: "¥xfe","¥xfe","¥xfe","¥xfe","¥xfe EXIT  ",
142: "¥xfeATDT 03-604-1111",   /* 電話番号をセット */
143: "¥xfe","¥xfe",
144: "¥xfe","¥xfe",
145: "¥xfe","¥xfe","¥xfe","¥xfe","¥xfe"};
146: 
147: void
148: init()
149: {
150:     int i,j;
151:     UBYTE fb[32];
152: 
153:     FNCKEYGT(0,fkeybuf0);
154:     for(i=0;i<20;i++) {
155:         for(j=0;j<32;fb[j++] = 0)
156:         strcpy(fb,fkeyname[i]);
157:         FNCKEYST(i+1,fb);
158:     }
159:     printf("ThunderTerm V0.01¥n");
160:     logfp = fopen(mktemp("ttXXXXXX"),"wb");/*適当なファイル名*/
161:     if (logfp == NULL) {
162:         printf("¥nログファイルを作れません¥n");
163:     }
164: }
165: 
166: void
167: finis()
168: {
169:     KFLUSHIO(0xff);          /* キーバッファクリア */
170:     fcloseall();             /* すべてのファイルを閉じる */
171:     FNCKEYST(0,fkeybuf0);/*ファンクションキーを元に戻す*/
172:     printf("Bye¥n");
173:     exit();
174: }
175: 
176: /* 最初に立っているビット番号を返す */
177: int
178: log(data)
179: int data;
180: {
181:     register int i,m;
182: 
183:     m = 1;
184:     i = 0;
185:     for(m=1,i=0;i<8;i++,m<<=1)
186:         if (m & data) break;
187: 
188:     return(i);
189: }
190: 
191: void
192: ulfile()
193: {
194:     char s[132];
195:     FILE *fp;
196:     int c;
197:     void wputrs();
198: 
199:     do {
200:         printf("アップロードするファイル名は:");
201:         scanf("%s",s);
202:     } while((fp = fopen(s,"rb")) == NULL);
203: 
204:     printf("¥"%s¥"をアップロードします。よろしいですか?(Y/N)",s);
205:     while(((c = toupper(getcon())) != 'Y') && (c != 'N'));
206:     if (c == 'Y') {
207:         printf("¥nＥＳＣキーで中断します¥n");
208:         while(!feof(fp)) {
209:             c = fgetc(fp);
210:             wputrs(c);
211:             if (isense()) {   /* RS-232Cのバッファサイズ */
212:                 c = getrs();   /* あるなら取り出す */
213:                 putcon(c);   /* 画面に表示する */
214:                 putchar(c);
215:             }
216:             if (check_esc()) {   /* ＥＳＣキーをチェック */
217:                 printf("中断しました¥n");
218:                 putchar(7);   /* bell */
219:                 fclose(fp);
220:                 return;
221:             }
222:         }
223:         fclose(fp);
224:         printf("¥nアップロードが終了しました¥n");
```

```
225:      } else {
226:         printf("中止しました¥n");
227:         return;
228:      }
229: }
230:
231: void
232: dlfile()
233: {
234:      char s[132];
235:      FILE *fp;
236:      int c;
237:      void wputrs();
238:
239:      do {
240:         printf("ダウンロードするファイル名は:");
241:         scanf("%s",s);
242:      } while((fp = fopen(s,"wb")) == NULL);
243:
244:      printf("¥"%s¥"へダウンロードします。よろしいですか?(Y/N)",s);
245:      while(((c = toupper(getcon())) != 'Y') && (c != 'N'));
246:      if (c == 'Y') {
247:         printf("¥nESCキーで中断します¥n");
248:         while(1) {
249:            if (isense()) { /* RS-232Cのバッファサイズ */
250:               c = getrs();    /* あるなら取り出す */
251:               putcon(c);    /* 画面に表示する */
252:               fputc(c,fp);
253:            }
254:            if (check_esc()) { /* ESCキーをチェック */
255:               printf("中断しました¥n");
256:               putchar(7);    /* bell */
257:               fclose(fp);
258:               return;
259:            }
260:         }
261:         printf("¥nダウンロードが終了しました¥n");
262:         fclose(fp);
263:      } else {
264:         printf("中止しました¥n");
265:         return;
266:      }
267: }
268:
269: /* 送信可能であることを確認して1バイト送信 */
270: /* ESCキーが押されたら return */
271: void
272: wputrs(c)
273: int c;
274: {
275:      while(1){
276:         if (check_esc())
277:            return;         /* ESCキーをチェック */
278:         if (osense()) {        /* 送信可能か? */
279:            putrs(c);    /* ならばRS-232Cに送る */
280:            break;
281:         }
282:      }
283: }
284:
285: /* ESCキーが押されたか? */
286: int
287: check_esc()
288: {
289:      return(BITSNS(0) & 2);
290: }
291:
```

リスト2　XMODEMファイル転送プログラム(XM0.C)

```
  1: /* XMODEMファイル転送プログラム : xm0.c */
  2:
  3: #include    <doslib.h>
  4: #include    <iocslib.h>
  5: #include    <stdio.h>
  6:
  7: #define    SOH    0x01
  8: #define    ACK    0x06
  9: #define    NAC    0x15
 10: #define    EOT    0x04
 11: #define    CAN    0x18
 12:
 13: int rsparam;    /* RS-232Cのパラメータ保存 */
 14:
 15: void
 16: main(argc,argv)
 17: int argc;
 18: char *argv[];
 19: {
 20:      int f;
 21:      FILE *fp;
 22:      char c0,mode;
 23:      void putrs(),init(),finis(),emptyrs();
 24:      void dl(),ul();
 25:
 26:      init();        /* X-on/off 解除 */
 27:
 28:      switch(argc) {
 29:      case 0:
 30:      case 1:
 31:         finis("パラメータが異常です");
 32:         /* break */
 33:      case 2:
 34:      case 3:
 35:         if (((c0 = argv[1][0]) != '/')  && (c0 != '-')) {
 36:            finis("第一パラメータが異常です。");
 37:            /* break */
 38:         } else if ((mode = toupper(argv[1][1])) == 'U') {
 39:            if (argc < 3) finis("ファイル名がありません。");
 40:            fp = fopen(argv[2],"rb");
 41:         } else if (mode == 'D') {
 42:            fp = (argc < 3) ? fopen(mktemp("xmXXXXXX"),"wb"):
 43:                  fopen(argv[2],"wb");
 44:         } else {
 45:            finis("第一パラメータの指定が'D','U'でありません。");
 46:            /* break */
 47:         }
 48:         break;
 49:      default:
 50:         finis("パラメータの数が多すぎます。");
 51:         /* break */
 52:      }
 53:
 54:      if (fp == NULL) finis("ファイルがオープンできません。");
 55:      emptyrs();    /* バッファクリア */
 56:      if (mode == 'U') {
 57:         ul(fp);    /* アップロード */
 58:      } else {
 59:         dl(fp);    /* ダウンロード */
 60:      }
 61:      finis("終了");
 62: }
 63:
 64: /* ダウンロード */
 65: void
 66: dl(fp)
 67: FILE *fp;
 68: {
 69:      UBYTE buf[256];
 70:      int f;
 71:      void wputrs();
 72:      int i,j;
 73:
 74:      wputrs(NAC);     /* 受信可能信号 */
 75:      while(1) {
 76:         f = getblock(buf,128+4);    /* 指定数のバイト数を得る */
 77:         if (f == (128+4)) {          /* 数が合っているなら */
 78:            f = check(buf,128+4); /* チェックサム等を確かめる */
 79:            if (f) {        /* 合ってる */
 80:               putchar('*');    /* 1ブロック終了合図 */
 81:               wputrs(ACK);    /* 受信確認 */
 82:               fwrite(&buf[3],1,128,fp);    /* 書き出し */
 83:            } else {        /* エラー */
 84:               putchar('X');    /* エラー合図 */
 85:               wputrs(NAC);    /* エラー信号 */
 86:            }
 87:         } else if (buf[0] == EOT) {    /* 送信終了? */
 88:            fclose(fp);
 89:            return;           /* 受信完了 */
 90:         } else {    /* バイト数が少ない */
 91:            putchar('x');
 92:            wputrs(NAC);          /* エラー信号 */
 93:         }
 94:      }
 95: }
 96:
 97: /* アップ・ロード */
 98: void
 99: ul(fp)
100: FILE *fp;
101: {
102:      UBYTE buf[256];
103:      int bl,i,sum;
104:      void wputrs();
105:
106:      while(1) {
107:         if (isense() && (getrs() == NAC))/*受信可能信号を待つ*/
108:            break;
```


▶ピンボーといわれたシャープ(MZ,CZ)ユーザーがなぜあんなにX68000を買えるのか疑問だった。が,その疑問が解けたとき私はX68000ユーザーになっていた(ついにEXPERT IIを買ったのでした)。同時に2年間のローンを背負っていた……。そ,そーだったのかぁっ(驚)。　升井　晋也(21)福岡県

```
109:     }
110:
111:     bl = 1;      /* ブロック番号 */
112:     while(!feof(fp)) {
113:        buf[0] = SOH;
114:        buf[1] = bl;
115:        buf[2] = ~bl;
116:        if ((i = fread(&buf[3],1,128,fp)) < 128) {
117:           for(;i<128;i++)/*128バイト以下なら残りを0で埋める*/
118:              buf[i+3] = '¥0';
119:        }
120:        for(sum=i=0;i<128;i++) {
121:           sum += buf[i+3]; /* チェックサムの計算 */
122:        }
123:        buf[128+3] = (sum & 0xff);
124:        do {        /* 送信 */
125:           for(i=0;i<128+4;i++) {
126:              wputrs(buf[i]);
127:           }
128:        } while(wgetrs() != ACK); /* 受信確認が得られるまで */
129:        putchar('*');
130:        bl++;
131:     }
132:     wputrs(EOT);
133: }
134:
135: /* データのチェック */
136: /* エラーが起きたなら0を返す */
137: int
138: check(b,n)
139: UBYTE *b;
140: int n;
141: {
142:     void error();
143:     int i,j,sum;
144:
145:     if (b[0] != SOH) {   /* ヘッダーのチェック */
146:        error("HEAD ERROR");
147:        return(0);
148:     }
149:     if ((b[1] & 0xff) != (~b[2] & 0xff)) {
150:        error("REC.NO ERROR");
151:        return(0);    /* ブロック番号のチェック */
152:     }
153:     sum = 0;
154:     for(i=3;i<n-1;i++) {
155:        sum += (b[i] & 0xff);
156:     }
157:     sum &= 0xff;
158:     return(sum == b[131]);    /* チェックサム確認 */
159: }
160:
161: /* エラーメッセージの表示 */
162: void
163: error(s)
164: UBYTE *s;
165: {
166:     printf("¥n%s¥n",s);
167: }
168:
169: /* 指定されたブロックをRS-232Cから読み出す */
170: int
171: getblock(b,n)
172: UBYTE *b;
173: int n;
174: {
175:     int i,j;
176:
177:     for(i=0;i<n;i++) {
178:        for(j=0;j<10000;j++) {
179:           if (isense()) {       /* RS-232Cのバッファサイズ */
180:              b[i] = getrs();    /* あるなら取り出す */
181:              break;
182:           }
183:        }
184:        if (j >= 10000) return(i); /* TIME OUT:バイト数を返す*/
185:     }
186:     return(i);    /* NO ERROR */
187: }
188:
189: /* 送信可能であることを確認して1バイト送信 */
190: /* ESCキーが押されたら終了する */
191: void
192: wputrs(c)
193: int c;
194: {
195:     void check_esc();
196:
197:     while(1){
198:        check_esc();          /* ESCキーをチェック */
199:        if (osense()) {         /* 送信可能か? */
200:           putrs(c);    /* ならばRS-232Cに送る */
201:           break;
202:        }
203:     }
204: }
205:
206: /* 1バイト受信 */
207: /* ESCキーが押されたら終了する */
208: int
209: wgetrs()
210: {
211:     void check_esc();
212:
213:     while(1){
214:        check_esc();          /* ESCキーをチェック */
215:        if (isense()) {         /* 受信可能か? */
216:           return(getrs());    /* ならばRS-232Cから得る */
217:        }
218:     }
219: }
220:
221: /* RS-232Cバッファをクリアする */
222: void
223: emptyrs()
224: {
225:     int i;
226:
227:     do {
228:        for(i=0;i<100;i++) {
229:           if (isense()) {          /* RS-232Cのバッファサイズ */
230:              getrs();
231:           }
232:        }
233:     } while(i<100);
234: }
235:
236: /* 入力可能か */
237: int
238: isense()
239: {
240:     return(LOF232C());    /* RS-232Cのバッファサイズ */
241: }
242:
243: /* 出力可能か */
244: int
245: osense()
246: {
247:     return(OSNS232C());    /* 送信可能か? */
248: }
249:
250: /* RS-232C読み出し */
251: int
252: getrs()
253: {
254:     return(INP232C());    /* 取り出す */
255: }
256:
257: /* RS-232C書き出し */
258: void
259: putrs(c)
260: int c;
261: {
262:     OUT232C(c);    /* あるならRS-232Cに送る */
263: }
264:
265: /* RS-232Cの初期化 */
266: void
267: init()
268: {
269:     printf("XMODEM V0.01¥n");
270:     rsparam = SET232C(-1);
271:     SET232C(rsparam & 0b1111110111111111);/* X-on/off 解除 */
272: }
273:
274: /* 終了設定 */
275: void
276: finis(s)
277: char *s;
278: {
279:     SET232C(rsparam);    /* 戻す */
280:     printf("¥n%s¥n",s);
281:     KFLUSHIO(0xff);          /* キーバッファクリア */
282:     fcloseall();        /* すべてのファイルを閉じる */
283:     exit();
284: }
285:
286: /* ESCキーが押されたか? */
287: void
288: check_esc()
289: {
290:     if (BITSNS(0) & 2) finis("ESC");
291: }
292:
293:
```

▶5月号のラジコンスティックを見て思い出したのが、3年ほど前アスカのターボファンを利用した小型ジェットエンジンを作った方がいた。「そいつはいいや。本当のジェットラジコン機が作れる。……しかしどうやって操縦するんだ」。　松永　直哉（21）東京都

PurePASCAL

新連載 PurePASCAL〈1〉

X68000にPASCALコンパイラを

Fujii Yoshimi　Fujiki Takeshi
藤井義巳/藤木健士

美しい言語PASCAL。この繊細かつ強力な言語をX68000で利用しようと開発したのがフルセットのPASCALコンパイラ"PurePASCAL"です。そして皆さんにもその魅力を味わってもらおうと入門講座を開始することになりました。

私たちがX68000のPASCALコンパイラを作ろうと思い立ったのは，いまから2年半ほど前のことです。その頃私たちは，大学でプログラミングを習っていました。私たちの学科は情報工学科だったので，プログラミングの学習に用いられる言語はPASCALでした（ほかの学科はFORTRANが多い）。

ところが，我らがX68000にはPASCALコンパイラがリリースされていなかったため，せっかくパソコンを持っていながら計算機センターに足を運んで演習問題を解いていました。ちょうどその頃，大学のコンパイラの講義で，PL/0というPASCALのサブセットを教材に，コンパイラの原理を学びました。以前からコンパイラには興味があったので，これを機会にX68000用のPASCALコンパイラを作ってみようと考えました。それが今回付録のディスクで配布するPurePASCALというわけです。

C言語に押されてすっかり影をひそめた感のあるPASCALですが，PASCALはとても美しい言語です。Cに比べたらずっと学びやすいため，初心者にも最適です。この機会にX68000ユーザーの皆さんもPASCALを勉強してみてはいかがでしょうか。

PASCALの生い立ち

PASCALは1960年代の終わり，スイスのチューリッヒ連邦工科大学(ETH)のNiklaus Wirth(ヴィルトと読む）教授によって考案されました。最初のPASCALコンパイラが動いたのは1970年のことです。その後，PASCALコンパイラはさまざまな計算機の上で実現され，多くのユーザーを獲得します。特にアメリカではC言語と同じくらい人気のある言語となりました。たとえば皆さんご存じのように，Macintoshの基本ソフトウェアは，PASCALプログラムから呼び出しやすいように作られています。

このように多くのユーザーに使われるようになると，言語仕様を規格化する必要が生じてきます。この要望に応えてISO7185標準PASCAL規格が作られました。

WirthはPASCALを設計するにあたって2つの点に重点を置きました。第1点はプログラミングの教育に適していることです。

PASCALが作られた時代はFORTRANの全盛期でした。FORTRANはデータ構造の記述力に乏しく，基本的なデータ構造（たとえば木構造やリストなど）を実現するにも配列を使って技巧的なプログラムを組まなければならないため，プログラミングの教育には適していません。こういった基本的なデータ構造を実現するために，PASCALはレコード，ポインタといった強力なデータ型を備えています。

また，プログラムの制御構造においても，FORTRANは再帰的なアルゴリズムを直接書き下すことができません。もちろんこの時代にはFORTRAN以外にも多くのプログラミング言語があったのですが，どれもWirthのお眼鏡にはかなわなかったようです。

第2点はコンパイラが作りやすいことです。当時，コンパイラはもっとも難しいプログラムのひとつで，職人芸的な作り方が行われていたようです。そのような状況ではせっかく素晴らしい言語を設計しても，コンパイラが普及しない可能性が大でした。WirthはPコードという中間コードを考案し，最初のPASCALコンパイラをPコードインタプリタ上で実現しました。作成が容易なPコードインタプリタを用意するだけでPASCALコンパイラを使えるようにするためです。

コンパイラの作成技術が進歩するにつれて，Pコードは使われなくなってきています（PurePASCALコンパイラはPコードを生成せず，直接AS.X形式のアセンブリコードを出力します)。

Cから見たPASCAL

数あるプログラミング言語のなかで，Cはソフトウェア開発の分野で現在もっとも広く使われている言語ではないでしょうか。皆さんの多くもXCやGCCを使ってプログラミングを楽しんだり，ソフトウェア開発のお仕事をされていると思います。

さて，その，皆さんよくご存じのCとPASCALとを比較してみましょう。それぞれの言語に長所，短所が存在しますが，基本的にはこの2つの言語は同じ系列の言語です。Cをある程度理解していれば，PASCALは簡単に覚えられると思います。では，それぞれの言語の特徴を述べる前に，2つの言語に共通した特徴を列挙してみましょう。

1）手続き型の言語である。

FORTRAN，PASCAL，C，Adaなどはすべて手続き型の言語と呼ばれています。手続き型以外の言語には，関数型，論理型などがあります。

2）構造化プログラミングをサポートする制御構造を備

えている。

構造化プログラミングをサポートするために，IF-ELSE，WHILE，REPEAT-UNTILなどの制御構造を備えています。

3) 豊富なデータ型を持っている。

整数型，実数型などの基本データ型に加え，複雑なデータ構造を実現するために，配列，レコード(構造体)，ポインタなどを備えている。

4) 関数，手続きは値呼び出し，参照呼び出しのどちらも可能である。

FORTRANは参照呼び出ししか許されていません。

そのほかにも共通点はあるかと思いますが，主なものはこの4点です。

では次に，PASCALに固有の特徴(長所，短所)を列挙します。

1) 分割コンパイルができない。
2) 型のチェックが厳しい。
3) 実行時チェックが行われる。
4) 関数，手続きのネスティングが許されている。

Cに対するPASCALの特徴で重要なのはこの4点でしょう。

1)は致命的ともいえるPASCALの欠点です。Wirthはプログラミングの教育には分割コンパイルは不要だと考えたのでしょうか。しかし，実際には多くの処理系において，分割コンパイルが可能なように言語の拡張が施されています。たとえばTURBO PASCALはVer.4.0から分割コンパイルが可能になっています。

PurePASCALコンパイラはVer.1.0では分割コンパイルをサポートしていません。独自の拡張を行うことによってプログラムの移植性を損なうことを恐れたからです。今後，ほかの処理系の動向を見極めたうえでどう対処するか判断したいと考えています。現在，Cで書いたプログラムとのリンクは制限付きですがサポートされています。

2)は意見の分かれるところだと思いますが，私はPASCALの長所だと信じています。Cは型のチェックが甘いために，思わぬバグをプログラムに潜ませる危険性があります。ANSI Cではこの点が見直され，プロトタイプ宣言などチェックのための機構がいくつか導入されました。PASCALは型のチェックが厳密に行われます。そのため，Cを使っている人の中には「PASCALは窮屈な言語だ」という人も多いのですが，結果的にどちらがわかりやすく，バグの少ないプログラムが書けるかはいうまでもないでしょう。

3)はCを使っている人には馴染みのない機能だと思います。簡単な例を挙げて説明すると，Cでは，

```
int a[10];
```

と定義された配列を次のような形でアクセスするとき，変数iの値が10以上だったり，負の値だったらどうなりますか？

```
a[i]=0;
```

運がよければバスエラーで止まりますが，多くの場合，配列の外の，関係のない領域を書き換えて知らんぷりで動き続けるでしょう。そして原因不明のバグの元となるのです。

PASCALはこのようなケースに適切なエラーメッセージを出してプログラムを中断させる機能を持っています。しかし，このようなチェックの機能はプログラムの実行速度にとってはマイナスです。PurePASCALコンパイラは，このチェック機能をオン/オフする機能を備えていますから，デバッグの段階ではチェック機能をオンにしておき，デバッグが終わってからオフにするようにすればよいでしょう。

4)はCとPASCALのプログラムの構造でもっとも異なる特徴です。CもPASCALも局所変数という概念は持っています。局所変数は手続きや関数の内部でだけ有効な変数で，ほかの手続きや関数で同じ名前の局所変数が宣言されても，これらは別々の変数として扱われるというものです。この考えを発展させPASCALではさらに，手続き，関数の中で局所的な手続き，関数が宣言できるのです。Cも将来，この機能を取り入れるかもしれませんね。

ところで，いまの4つのなかにはありませんが，PASCALには「手続き」という単位があります。これは値を返さない関数と考えればよいでしょう。Cでいうvoid型の関数にあたるものです。

PASCALでプログラムを書いてみる

前置きが長くなりましたが，プログラミング言語を覚える最良の方法は，その言語を使ってプログラムを書いてみることです(もはやこれは常識！)。早速PASCALでプログラムを書いてみましょう。Cの教科書「K&R」でいちばん最初に出てくるプログラムは“hello.c”でしたね。

```
/*'hello.c' */
#include <stdio.h>
main()
{
    printf("Hello, world¥n");
}
```

確かこんなプログラムだったと思います。これと同じことを行うプログラムをPASCALで書いてみます。

```
 (*'hello.pas'*)
program hello(output);
begin
    writeln('Hello, world')
end.
```

どうですか，細かい違いに目をつぶれば，実によく似ていると思いませんか。Cのプログラムは‘main’という名の関数から実行が始まります。それに対してPASCALのプログラムは‘program’で始まり，‘.’で終わります。‘begin’と‘end’はそれぞれCの‘{’と‘}’に対応しています。

プログラムの最初の1行はどちらもコメントです。‘writeln’という標準手続きは印字と改行を行います。writeln('Hello, world')の後ろにセミコロンがないのはタイプミスではありません。CとPASCALではセミコロンの役

割が少々異なるのですが，この場合，セミコロンをつけても空文と見なされるだけで，ちゃんと動きます。

プログラム名('hello')は自由につけることができます。プログラムの引数は，入出力のファイル変数を表しています。'hello.pas'では標準出力'output'のみを使用すると宣言されています。

次に，もう少し大きなプログラムを書いてみます。リスト1をご覧ください。行番号は説明のためにつけたものです。

さて，プログラムの解説を始めましょう。今月は細かな言語仕様については触れません。「PASCALのプログラムはこんなものなんだなあ」と，雰囲気をつかんでもらいたいと思います。細かい点については来月以降順を追って説明していくつもりです。

'sieve.pas'では標準入力と標準出力(それぞれ'input','output')を使用します。どこで使用しているかといいますと，標準手続き'readln','write','writeln'で暗黙に使用されているのです。プログラムの名前はCにならって'main'にしました。'{'と'}'で囲まれた部分は'(*'と'*)'と同じくコメントです。ただ，'{'と'}'は，PurePASCALコンパイラではコンパイラへの指示を与えるのにも使われます。

もう気づかれた方も多いと思いますが，メインルーチンの本体はいちばん最後にあります。これもPASCALの特徴です。「すべての名前は使われる前に宣言されなければならない」というのがPASCALの考え方ですので，メインルーチンで使われる手続き，変数，その他はすべてメインルーチンよりも前に置かれるわけです。言い換えると，メインルーチンが最後に置かれることになります。

4，5行目は定数宣言です。Cのマクロのとは違って，純粋に定数に名前をつけただけのものです。式は書けません。

6～8行目はグローバル変数の宣言です。変数の宣言は'var'で始まります。ここで宣言した変数はプログラム中のどこからでも利用できます。

7行目の配列の宣言はCとはだいぶ違っているでしょう。PASCALの配列はCのように下限が0とは決まっていません。下限も上限も自由に設定できます。なおこの配列の要素は論理型です。Cは整数型で論理型を代用していましたね。PASCALの論理型は'true'か'false'の値をとります。

10～42行目は手続きの宣言です。これらの手続きは，メインルーチンから呼び出されています。それぞれの手続きの中でローカル変数が宣言されているのがわかると思います。引数の宣言方法もCと比較してみてください。ANSI Cの引数宣言によく似ているでしょう。

プログラムを読んで気づいたと思いますが，PASCALでは代入の記号に'：='(becomes)が使われます。Cに慣れた人はしばらくのあいだは間違うでしょう。また，関係演算子の等号には'='が使われます。Cと違って代入は文であって，式の要素にはなりえません。Cでよく条件式の中で'='と'=='を間違えたりしますが，PASCALではそのようなことはありません。

';'の使い方がCとは少し違うと書きましたが，どう違うかわかりますか？ Cでは';'はひとつの宣言の終わり，文の終わりを示します。PASCALもほとんど同じに見えるのですが，'end','else'の前には';'はつきません。このことから，PASCALの';'は，文と文の区切りと考えられます。それでも'end'の前の余分な';'は空文と見なされるだけで支障はありません。たとえば，

```
begin
    a：=b;
    b：=c;
end
```

は次のように考えられます。

```
begin
    a：=b;
    c：=c;
     (*  空文  *)
end
```

それに対して'else'の前に余分な';'を置くif文がそこで終わったと見なされ，文法エラーになりますので注意が必要です。次のプログラム片を見てください。

```
if a > b then
    x：=a;
else
    x：=b;
```

コンパイラは'else'のところで構文エラーを検出するでしょう。なぜなら'x：=a;'のところでif文と次の'else'で始まる(間違った)文が区切られたことになるからです。

もうひとつの例を見てください。

```
if a > b then
    if a > c then
        x：=a (*';'がないことに注意*)
else
    x：=c;
```

インデントから察するに，このプログラムの作者は「aがbより小さかったらxにcを代入し，aがbより大きかったらaとcと比較し，大きいほうをxに代入する」つもりだったようです。しかし，作者の意図に反してこのプログラムは，「aがbよりも大きかったらaとcを比較し，大きいほうをxに代入する」プログラムだとコンパイラは解釈します。インデントを変えて書き直してみるとわかるでしょう。

```
if a > b then
    if a > c then
        x：=a
    else
        x：=c;
```

なぜこのようなことが起こるのでしょうか。それは「'else'はもっとも近い'if'に対応する」という規則によります。このことはC言語でも同じようにいえます。この種のバグがもっとも見つけにくい種類のものであることは，皆さん十分に経験ずみでしょう。もし最初に意図した動作を

させようと思ったら，'begin'と'end' を使って次のように書けばよいでしょう。

```
if a > b then
  begin
    if a > c then
        x：＝a
  end
else (＊この'else'は最初の'if'に対応)
    x：＝c;
```

今回のプログラム'sieve.pas'は，コメントにもあるように,「エラトステネスのふるい」を使って素数を計算します。実際にPurePASCAL コンパイラにかけて動かしてみてください。また，Cで同じことをするプログラムを作って比較してみるのもよいでしょう。来月からの本編に備えてPASCALコンパイラに親しんでおいてください。

なお，今回のコンパイラで作成したプログラムについてですが，ロイヤリティなどという面倒なものは必要ありません。このコンパイラを使ったということをひと言書いていただければそれで結構です。あとはご自由に使用してください。

また，使用してみて動作がおかしいなどの場合はOh!X編集部まで連絡してください。

〈参考文献〉

[1]Jensen, K. and Wirth, N.,「PASCAL-User Manual and Report, 3rd edition」, Springer Verlag, 1985
邦訳(原田賢一),「PASCAL」, 培風館, 1988

プログラミング言語PASCALのISO規格に基づく報告と，解説が載っている。PurePASCAL の開発の際，規格書として用いた。前半の解説はPASCALの入門書としても最適である。

[2]石畑，寛，安村,「Pascalの標準化 －ISO規格全訳とその解説－」, bit別冊，共立出版, 1984

こちらはISO7185規格の全訳とその解説が載っている。PASCALの厳密な規格が知りたい人はこちらをお勧めする。解説の部分では，規格の不備の補完や概念の説明がなされている。PASCALについてより詳しく知りたい人はご一読を。

リスト1

```
 1: (* 'sieve.pas' *)
 2: (* エラトステネスのふるい *)
 3: program main(input, output);
 4: const
 5:     MAXNUMBER = 100000;
 6: var
 7:     s:array[1..MAXNUMBER]of boolean;
 8:     max:integer;
 9:
10:     procedure initialize(max:integer);
11:     var
12:        i:integer;
13:     begin
14:        for i:=1 to max do
15:           s[i]:=true
16:     end; { initialize }
17:
18:     procedure display(max:integer);
19:     var
20:         i:integer;
21:     begin
22:         for i:=2 to max do
23:             if s[i] then
24:                 write(i:6);
25:         writeln
26:     end; { display }
27:
28:     procedure sieve(max:integer);
29:     var
30:         i, j:integer;
31:     begin
32:         for i := 2 to round(sqrt(max*1.0)) do
33:             if s[i] then
34:             begin
35:                 j := i * 3;
36:                 while j <= max do
37:                 begin
38:                     s[j] := false;
39:                     j := j + i
40:                 end
41:             end
42:     end; { sieve }
43:
44: begin { main }
45:     writeln('Input a number(1..', MAXNUMBER:0, ') please.');
46:     readln(max);
47:     initialize(max);
48:     sieve(max);
49:     display(max);
50: end.  { main }
```

PurePASCAL ◆

追跡レポート　これがSX-WINDOWだ(2)

未来を語るアクセサリ

Yoshida Kouichi
吉田　幸一

X68000の新たな環境として注目を浴びているSX-WINDOW。今回はシステムに付属しているアクセサリ類を1つひとつ紹介しよう。そこには今後実現されるかもしれないウィンドウ上のアプリケーションの姿を垣間見ることができるだろう。

SX-WINDOWというのはウィンドウシステムだから，SX-WINDOW用にプログラムを書くと，SX-WINDOW上で動く。これが基本だ。とはいえ，まだやっとシステム自体(Ver.1.0)ができたばかりで，対応するアプリケーションはほとんどない。残念といえば残念である。とても残念である。

だが，しかし，標準でついてきた少数のアプリケーションは未来を語るもの（語るだけであって，未来そのものではないぞ）であった。これが重要である。そんなに数はないので，順番に見ていこう。

その前に，2つのタイプに分類しておこう。外部ファイルの入出力を伴うアプリケーションと，そのプログラム独立で動作するアプリケーションである。

データを必要とするアプリケーション

先月号でファイルのタイプやダンプがポップアップメニューでできるという話をしたが，これはようするにタイプ.Xやダンプ.Xというプログラムにそのファイルを渡して実行しているだけのことだったりする。

●タイプ.X，ダンプ.X

というわけで，タイプ.Xを実行してみよう。すると，びよんとタイプのウィンドウが現れる。タイプすべきファイルが指定されていないのだから，中は真っ白である。

たとえばビジュアルシェルだったらどうしただろうか，てのを考えてみると，容易に，“アイコンの実行ファイルにタイプコマンドを指定する”というのが思い浮かぶだろう。もちろん，SX-WINDOWでもこれは可能である。が，そんな不自由なことをしなくてもよい。

SX-WINDOWのタイプ.Xのウィンドウは，“いつでもタイプしたげるよん”と，ファイルを待っているのだ。

そこで，私はタイプしたいファイルのアイコンをドラッグして，タイプウィンドウの上へと運び，指を離す。すると，タイプウィンドウにそのファイルの内容がタイプされるのだ。マルチウィンドウだから，タイプウィンドウが隠れていることもあるだろう。それでも僅かでも見えていれば受け取ってくれるのである。

つまり，アプリケーションにあとからデータを渡すことができるのである。さらに，何かを表示しているタイプウィンドウに別のファイルを放り込むと，古いファイルは捨てて，新しく放り込んだファイルをタイプしてくれる。ダンプ.Xも同様。

これはなかなか新しい。アプリケーションを実行してもウィンドウを開くだけで，そこにデータを放り込んで初めて仕事を始めるのだ。

もちろん，ウィンドウの大きさは可変だし，いくつもウィンドウを開けられるので，2つのファイルをダンプして比較したりするのも簡単である。

タイプやダンプでのポイントは，右上のプリンタアイコンである。ファイルの印刷はタイプやダンプウィンドウから行うことができる。

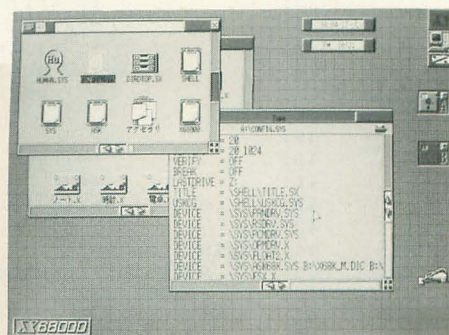
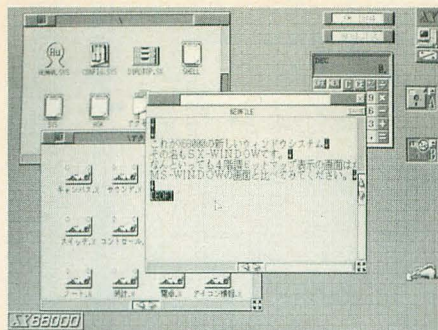
●ノート.X

タイプだけあってもしかたがないわけだが，それでもまともなウィンドウエディタは間に合わなかったみたいで，とりあえずついてきたのがノート.X。ノートといってもビジュアルシェルでついてきた罫線の入ったノートではなくて，簡易エディタである。現段階ではコメントを差し控えたい，としかいいようがないな。CONFIG.SYSやバッチファイルの編集くらいには使えるけれども。SX-WINDOWが市販されるころには（SX-WINDOWはいまのところ，X68000EXPERTIIかPROIIを買った人しか持ってなかったりする）もっとよくなっているはずであるから，そのときに語ろう。

なお，ノートでもタイプやダンプと同様，開いたウィンドウにファイルアイコンを放り込んでやるとそいつが編集できる。

●キャンバス.X

タイプやダンプだけが表現ではない，文字だけが見られたって，それがどうしたってんだ。てなわけで，キャンバス.Xは**グラフィック観賞ウィンドウ**を開くプログラムである。拡張子が.PIXのファイルをダブルクリックすると，（それが拡張子の名に恥じぬグラフィックデータだったなら）その絵の大きさでウィンドウが開く。768×512ドットモードオンリーのウィンドウシステムなので，グラフィックも768×512の解像度で65536色中16色のものに限られる。グラフィックは当然グラフィック画面に描き，テキスト画面を使ったウィンドウはただ絵のある範囲だけ抜いてやればいいわけで，単色

ファイルアイコンをマウスカーソルでつかんでタイプウィンドウに放り込むと……。

ハイ，このとおり。TYPEコマンドと同じく内容表示されるのです。

ノート.Xはごく簡単なエディタだ。やはり，ファイルを放り込むと編集できる。

グラフィックを表示するキャンバス.X。メモリさえあれば何枚でも開けるぞ。

サウンド.XのウィンドウもOPMファイルやPCMファイルを待っている。

電卓.Xと時計.Xだ。これまた，いくらでも開けるぞ。

4階調のウィンドウ上に16色とはいえカラーのグラフィックがぽつんとある図はなかなか違和感があって面白い。

キャンバスはグラフィックのタイプコマンドのようなもので，いくつも開くことができる。ファイルアイコンをドラッグするとそのデータがキャンバスに描き込まれる。**マルチメディアへの道第1号**てなわけだ。

グラフィックのパレットは最も上にある絵のものになるため，いろんなウィンドウの下敷きになったキャンバスの絵は変な色に染まったりして，また，楽しいものではある。

●サウンド.X

マルチメディアへの道第2号がこのサウンド.Xである。もうおわかりのとおり，OPMファイルやPCMファイルを鳴らすためのコマンドである。ただし，一度にひとつしか開くことができない。OPMもPCMも含めて，である。

サウンドプレイヤーには操作パネルとスピーカーの絵がある。拡張子がOPMのファイルをダブルクリックするなりサウンドプレイヤーのウィンドウに放り込むなりすると，OPMファイルの演奏が始まる。スピーカーの間にはOPMファイルのアイコンと同じシンセサイザマークが入り，ファイル名が表示される。操作パネルにはプレイ，ポーズ，ストップだ。また，拡張子がPCMの場合は，PCMファイルの演奏(?)が始まる。操作パネルにはプレイとストップしかない。

拡張子がOPMでもPCMでもないアイコンをドラッグしてやるとどうなるかっていうと，PCMかOPMか聞いてくるのである。PCMファイルはこうでなければ音が鳴らないというフォーマットがない（ヘッダとかデータ構造とか）ので，Human.SYSを鳴らすなんてこともできる。

●アイコン情報.X

アイコンデータを見たり変更したりするためのコマンドである。が，アイコンのデザインを変更したり，新しいアイコンを作ったりはできない。とりあえず，アイコンをアイコン情報ウィンドウに持っていくと，ダブルクリック時の実行ファイルやオプションを変更できる。

*

とまあ，こんな感じである。ポイントは，**データファイルが渡されるのを待っているウィンドウがいくつも開ける**ことにつきる。この方式はどんどん想像力の翼を広げて損のない方式だ。

たとえばFC.Xだと，“実行するとファイルアイコンを渡すところが2つあるウィンドウが開き，比較したいファイルを2つ放り込んだら起動してファイルの比較をしてくれる”という姿が容易に想像できる。

データが来るまで実行スタンバイしてるウィンドウなんて，なかなか素晴らしいではないか。きっとコンパイルウィンドウにファイルアイコンを入れるとコンパイルしてくれたり，印刷ウィンドウにワープロのテキストファイルを放り込むと印刷してくれたり，ワープロウィンドウに同様にファイルを放り込んだりといろいろ考えは広がるのであった。

独立して動くアプリケーション

で，そのほかのアプリケーションについても紹介しておこう。

●電卓.X

16進演算機能付きの電卓である。もちろん，電卓を画面に10個並べることだってできる。計算結果をクリップボードに入れられるのはなかなか。

●時計.X

時計である。この時計は，いわゆる**疑似マルチタスク**ってやつで動くように作られている。つまりだ，時計は単なるアプリケーションのひとつに過ぎなくて，時計を10個出せば，10個の時計が並行して動くのだ。難をいえば，デジタル時計で字が小さい，ってことだろうか。なお，この時計にはスイッチがあり，時刻表示と日付表示が切り換わる。通常は時刻表示の時計.Xと日付表示の時計.Xのふたつを画面の好きなところへ置いておくことになるだろう。時刻のアジャストはコントロール.Xで行う。

●コントロール.X

俗にいう，コントロールパネルである。コントロールパネルウィンドウが開くと，中にはさらに7つのアイコンがある。どーでもいいけど，コントロールパネルもちゃんとしたウィンドウであるから，何枚開いてもよい。

7つとは，ディスプレイ，キーボード，マウス，プリンタ，スイッチ，RS-232C,

SX-WINDOWは資源を食う

ちょっと想像すればわかるとおり，プログラムサイズはとても大きい。とりあえず，2Mバイトのメインメモリがないと動かないので，覚えておくように。何かを常駐させたりという特別なことはしていないのだが，メインメモリ2Mバイトの状態で，971Kバイトしか残らなかった。これは開いたウィンドウや動かしているプロセスの数で変わるのだが，これを見ただけで2Mバイト必要なんだな，ということがわかる。ついでに，そんな状態だからRAMディスクも確保は難しいわけで，ハードディスクは欲しい（最近，何を読んでもこればっか）。

さらに，わがままをいうと，せめて16MHzの68000を使うか，68020を使った贅沢機種を安く作ってほしいぞ。

しかし，**コンピュータにとって最も重要な資源は人間**である。このことを忘れてはならないと思う。

タイマである。設定を見たり変更したいアイコンをダブルクリックすればそれぞれのパネルが開く。これはあくまでもパネルであってちゃんとしたウィンドウではない。

たとえば，ディスプレイでは表示色が変えられる。マウスではMacintoshばりに移動速度やダブルクリック感応速度が変えられる。タイマは，あのビジュアルシェルのものと同じ（ビジュアルシェルのタイマの操作性は先進的でない)。スイッチは電源OFF時の処理や文字出力（￥を＼にしたりとか）がある。

●スイッチ.X

コントロールパネル以外のスイッチがここにある。SWITCH.Xが2つに分かれたというわけね。

スイッチ.Xで開くのはちゃんとしたウィンドウではなくてパネルである。2つも3つも開いたり，移動したりはできない。それは設定する内容の問題である。メモリ容量とか，つながってるハードディスクの数とか，SRAMの用途とか，起動時のドライブ設定とかを決めるのだ。

もちろん，マウスで簡単操作。

●背景設定.X

最近のウィンドウシステムでは背景パターンを変更できるのが流行らしい。NeXTでもそうだったし，ハイパーワードでもできた。SX-WINDOWでもできる。パターンが8つまで登録されていて，そのうち6つはユーザー変更可能なのだ。試しにアンク十字架を作って背景にちりばめてみたら，結構不気味だった。4階調あるので，結構凝った背景も可能だ。

●カレンダー.X

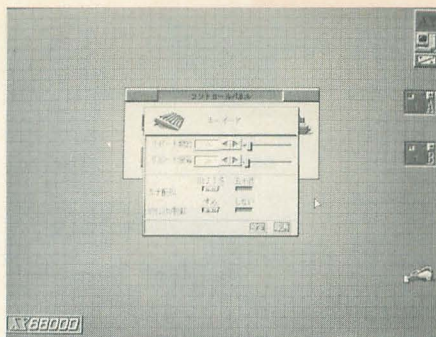
えっと，カレンダーである。それだけ。

＊

以上で，システムディスクのアクセサリディレクトリに入っているコマンドは終わりである。あとはデータディスクにピンボールと暁子がいるだけである。

●ピンボール.X

初代X68000にはグラディウスがついてきた。SX-WINDOWにはピンボールがついてくる。テキスト画面を使っているので，単色4階調である。が，これがなかなかセンスがよい。マウスの左右のボタンで遊ぶ。

ポイントは2つ。ひとつは，ちゃんとしたウィンドウであること。だから，好きな場所で遊べるし，ピンボールウィンドウを開いておいて，やりたくなったらピンボールをアクティブウィンドウにするなんてこともできる。ちなみに，マウスカーソルがピンボールウィンドウ上にないと操作できないがボールは動いていたりする。

コントロール.Xでは表示色やマウス，キーボードなどの設定が細かく変えられる。

もうひとつは，ピンボールウィンドウとスコアウィンドウの2つのウィンドウが開いて，それぞれ独立したウィンドウのように扱えるのだが，互いにリンクしている。スコアウィンドウがいろんなウィンドウの下敷きになっていてもピンボールウィンドウがアクティブなら遊べて，なおかつ下敷きになったスコアウィンドウでは人知れずスコアがカウントされていたりするのだ。こいつは応用がききそうである。互いに独立したウィンドウとはいっても，片方だけクローズしたりはできない。ポイントはひとつのアプリケーションが複数のウィンドウを持てて，互いに通信してる，てなことだ。

ちなみに，スコアウィンドウにはでっかくSPSと描いてあったりする。

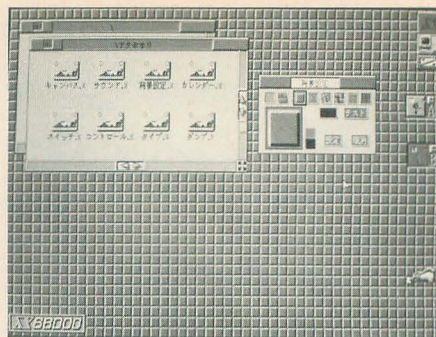
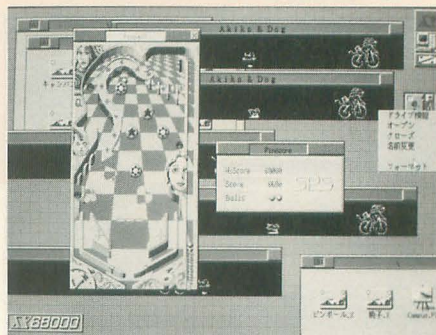
●暁子.X

どーでもいいけど，暁子が辞書に入ってなかったぞ。で，この暁子.Xは疑似マルチタスクサンプルとして入っているだけなのであった。

発展的なまとめ

よく見てみると，全アプリケーションにひとつの共通点がある。それは，**プログラム名が日本語**なことだ。ウィンドウシステムだから起動時にファイル名を入力する必要がない→打ちやすい英語でなくてもいい→わかりやすい日本語だ！　てなわけである。ここで，MS-DOSより長い名前をつけられるHuman68kのおいしさが現れてくる。

アイコンだけでなく，それを援助するわかりやすいプログラム名。そのタッグが重要。Macintoshだってあの使いやすさの裏には長いファイル名がつけられる点があることを見逃してはいけないのだ。

データファイルだって重要で，手で打たなくてもいいなら，わかりやすい日本語であるに越したことはない。たった8バイトではそれも難しいが，Human68kは18バイトなのだ(識別は最初の8バイトだけなのだが)。MS-DOSマシンで読む可能性のあるファイル以外は，日本語で長い名前をつけよう。

背景設定.Xで背景のパターンを自由に変更できる。好みのパターンに書き換えよう。

結構遊べるピンボール.Xと疑似マルチタスクのサンプル暁子.Xだ。

さて，冷静に考えてみると，ついてくるアプリケーションが完璧というわけではない。時計だってアナログなやつが欲しいし，カレンダーなんて絶対手抜きだ。ノートに至っては何をかいわんや。しかし，それは決定的な欠陥ではない。ここも重要なところで，ビジュアルシェルは閉じた世界だったからアクセサリが悲惨だったらどうしようもないのだが，SX-WINDOWは開かれたシステムだから，気に入らないアプリケーションがあれば作ってしまえばいいのだ！　アナログ時計なんて絶対誰かが作るさ。PDSだってたくさん出回るさ。

というわけで，カレンダーだって，きっと日付をダブルクリックしたらその日のスケジュールが起動するようなものを作ればいいのだ。だから，SX-WINDOW開発キット＆資料が出るまで，SX-WINDOWはビジュアルシェルのようにしか使えないかもしれないけれど，将来性に期待しようではないか。

ウィンドウシステムのいいところは，小さなアプリケーションの積み重ねで大きな環境ができるところにある。

電子手帳とリンクするソフトを作りたいと思ったら，従来なら（普通のエンドユーザーでも使えるように）転送されたデータ

を活用する部分からデータのやりとりまで全部作る必要があったが，ウィンドウシステムなら電子手帳のデータをCSV形式にコンバートするウィンドウさえ作れば（クリップボードに入れてもいい），あとはそのデータを別のウィンドウに渡してやればいいのだから。素人プログラマの腕の見せどころ，てなわけだ。

というわけで，私は楽観的なのである。早く開発セットが出ないかな，と。

ウィンドウシステムに対応するために

ウィンドウシステム。これは新しい。少なくとも，従来のコマンドシェル思考が通用しない面がたくさん出てくる。そいつを頭に入れておかないと，結局，慣れた'80年代の環境に閉じ込もってしまうので注意だ。そうならないように，思いつくままに書いておくので，読むように。

1）ファイル管理

コマンドシェル環境とウィンドウ環境。ファイルの管理の仕方が変わるだろう。たとえばディレクトリだ。いままではファイルの位置と名前さえ覚えていればよかったが，ウィンドウシステムでは見たいファイルのあるディレクトリウィンドウを開かねばならない。すると，よく使うデータが深いサブディレクトリにあったりすると探すだけで面倒になってしまう。

ディレクトリ管理やファイル管理の仕方というより考え方が変わってくると思う。今度Macintoshユーザーにでも聞いてみよう。

2）バッチファイルはない

PC-9801ユーザーがMacintoshユーザーにバッチファイルについて説明するのに苦労した，てな話を読んだことがある。バッチファイルはMacintoshにはないのだ。つまりはそういうことだ。

ウィンドウシステム用簡易バッチなんてのは簡単に作れそうだけれど，それがいいことかどうかはわからない。もしかしたらウィンドウシステムというのは，バッチファイルなんていらない文化かもしれないからだ。

ウィンドウシステム下のバッチは，一度に複数のウィンドウを開くようなものかもしれない。複数のアプリケーションを同時に立ち上げられるのだから，バッチのような逐次処理しなくてもすむことはしなくてもいいのだ。

3）オプション付き起動もない

コマンドシェル文化の育てたものにオプション付き起動がある。バッチファイルやエイリアスも重宝された。だが，ウィンドウシステム文化ではオプション付き起動文化は廃れていくだろう。新しく，ウィンドウを開いてから必要な設定をメニューでするような，そんな文化だ。

4）アプリケーション管理

というわけで，これまではパスを切ったり，バッチファイルにして専用のディレクトリに収めたり，エイリアス機能を使ったりすれば，プログラムのある場所を意識しなくてもいつでも実行できた。ウィンドウシステムではそれができない。だから，（特にハードディスク使用時の）アプリケーション管理も変わるだろう。

5）その他

いままでは統合ソフトといえばでかいと相場が決まっていた（kamikazeなんか，500Kバイト以上ある）。そのうえ，1本のソフトが多くの機能を持つ必要があった。しかし，ウィンドウシステムなら，ワープロと表計算と通信とデータベースを一度に立ち上げることができるのだ。これはつまり，バラバラのソフトがあれば統合できるということだ。実際には，768×512ドットの小さな画面ではそんなことは実用的ではないし，結局画面一杯のウィンドウを開いてしまう大きなソフトが出てきてしまうだろうけれども（Macintoshを見てるとわかる），わざわざひとつのソフトを終了させなくても別のソフトに触れられるおいしさは残るはずだ。エンドユーザーにとって，これはとてもおいしいことではないか。

てなわけで，いまX68000ユーザーのほとんどはMS-DOSの育てたコマンドシェル文化のただ中にいる。その感覚にとらわれている限り，すぐ画面が埋まってしまってブラインドタッチより反応の遅いウィンドウシステムがいいとは思えないだろう。便利なアプリケーションが揃うまで，ウィンドウシステムは遅くてうっとうしいとしか思えないだろう。

しかし，マウス1個でどこまでできるかに挑戦するウィンドウ文化はまぎれもなく，新しいパソコン文化への道なのだ。ウィンドウシステムのマンマシンインタフェイスはまだまだ過渡期で，どうすれば使いやすいものができるかは，わかってないのだ。少なくとも，ウィンドウシステム上で複数のアプリケーションが同時に立ち上がるということは，データや操作性統一への第一歩である。私は期待している。まだどうなるかは皆目見当もつかないけれども，アプリケーションごとに異なった画面構成やら操作性を覚えるのは嫌だし，キーボードに頼った入力も嫌だし，いちいちひとつのソフトを終わらせて，別のを立ち上げなければならない面倒臭さも嫌だから。

なんか，また能書きが多くなってしまったな。ま，いいか。

コマンドシェルウィンドウは必要か？

すでにいろんなところで聞いている。

「SX-WINDOWにコマンドシェルウィンドウがないのが残念だ」という声だ。

確かに，ウィンドウシステム内でコマンドシェルがいくつも動いたら（OS-9やX-WINDOWみたいに）便利だろう。それはわかる。しかし，私はその意見に否定的な見解を示したい。

理由1

ウィンドウシステムではどう考えても画面一杯使ったコマンドシェルよりスクロールなんかが遅くなる。コマンドシェル文化に染まった人が，その速度に耐えられるわけがない。

理由2

ウィンドウシェルがあると，本文で書いたように，新しいウィンドウ文化（そういうものがあればだけど）が育たない。みんな，慣れた環境がいいに決まっている。

理由3

その前にやることがある。つまり，そういう注文は，ウィンドウ上でのさまざまな環境が整ってからいってもいいんじゃないかということだ。エディタやウィンドウ上での開発ツールができて，それらを使ってみて，それでもコマンド環境が欲しいと思ったときに作ればいい。初めからコマンドシェルウィンドウがあったりすると，結局，コマンド文化から進歩しない気がするのだ。

確かに，慣れたユーザーにとっては，マウスを動かすよりキーをタイプしたほうがずっと速いのはわかっている。私だって，SX-WINDOWが単なる高級なビジュアルシェルにすぎないのであれば，コマンドシェルを使う。ノウハウもたまっているし。

しかし，キーボードに頼るのは嫌だし，なんといっても，複数のアプリケーションが同時に立ち上がるのは魅力だ。初心者にあの殺風景なコマンドシェルを教えるのも面倒臭い。

MS-WINDOWSはコマンドシェル上で立ち上がるアプリケーションの共通の基盤としてのウィンドウシステムであった。だから，コマンド文化を引きずっている。だから，コマンドシェルに慣れたユーザーでも，必要に応じて使うには（アプリケーションさえ揃っていれば）便利な代物だ（その割に大きな資源を要求するけれど）。でも，SX-WINDOWはMacintoshみたいにそれ自体でコマンドシェル環境も吸収してしまおうという，無謀なほど壮大なものだ。無理をしすぎてコケないようシャープさんには気をつけてもらいたい。

そんでもって，ソフトハウスさんもできるものならウィンドウから起動してウィンドウに帰って来るような，そんでもってハードディスクにすぐ組み込めるような（ゲームも含めて）ソフトを作ってもらいたいものである。

〔第1回〕

Oh!Xアンケート結果大分析大会

解説/荻窪 圭

2月号で実施した愛読者アンケートの集計結果が出ました。パソコンの具体的用途から，プログラミングのレベル，周辺機器やソフトの保有状況，そして掟破りのベストライターまで，一挙に公開（ちょっと後悔）しちゃいましょう。

本日は，創刊8周年記念のOh!Xをお買い上げいただき，どうもありがとうございます。うーん，それにしても8年ですか。8年。創刊時に大学に入った人は，もう留年できない年。卒業できなきゃ横から出るだけ。昔，大学のSF研にいた頃，エイトネンという歌が流行った(SF研は私も含めて留年生の宝庫だったから)。どっかの大学のSF研の人が作ったらしいけど，エイトマンの替え歌で，“落ちるしーけん，落ちるつーいし，落とすたんーい……残れエイトネーン，誰よりもなーがくー……”という歌詞だった気がする。あ，関係ない話か。

閑話ついでにいっちゃうと，私の知り合いには8年間大学生してたやつがいるけど，ちゃんと卒業したし就職したし結婚もした。だから，8年っていってもそれほど長い気がしないのだ。八は末広がりっていうくらいだから縁起もいいことだし。

縁起ついでに，ここに2月号で行った大アンケートの集計結果（抜粋）を発表することになった。案内役は私，“午前2時40分の男”荻窪圭と，アシスタントはX68000唯一の本格的表計算ソフト，スクロールは遅くても心は錦の“Kamikaze”である。スチャラカチャンチャン。

1 Oh!X読者はパソコンで何をしてるか

さっそく2月号のアンケート結果発表である。ちなみに，無作為抽出300枚のデータを元にしているので，アンケートを送ったのに反映してない！ てな人もいるだろう。でも，視聴率なんて何千万のテレビのうち，数百台しかサンプリングしてないしね。そういえば，3月20日の朝日新聞にアンケートに関する面白い記事が出ていた。日本原子力研究所労働組合が職員約1800人にアンケートを実施し，約350人から回答を得たのだそうだ。で，日本の原発の事故の可能性について302人が回答し，細かい数字は抜きにして，事故は起こらないと回答した人より一抹の不安があると答えた人のほうが多かったそうである。で，日本の原子力開発の現状について6割が「危機的状況にある」と答えたそうだ。この結果も重要だが，しょってるのはそれに対する日本原子力研究所人事部長の談話だ。彼は原研職員は約2500人いるが，回答者は約300人なので，結論を出すにはデータが少なすぎる，というようなコメント。

ここで私は悩んだ。フツーに考えて，2500人中300人というのは，十分有意な数ではないだろうか。やはり権力はあやしい。

というわけで，すべての調査というものは「サンプリングが少なすぎる」といってしまえばいえるものなのだ。視聴率だって，偶然，機械の置いてある家庭では誰も見てなくて，そうでない家では100万人が見ていたかもしれない番組だってあるのだ。でも，それでも長い目で見れば有意なのだ。信じるものは救われる。どこを救われるかっていうと，たいていの場合足元だ。

話がそれてしまった。で，300枚である。そのうち，XでもMZでもないユーザーのものが8枚あった。よって，表によっては292枚をサンプル数とした値が出ていることがある。

ちなみに，X68000ユーザーは151人。

では表1である。値はすべて％である。グラフ化したのがグラフ1。当然重複（どーでもいいが，ちょうふく，と読んでください）解答がたくさんあるので，合計すると何百％にもなる。

表1 パソコンの用途

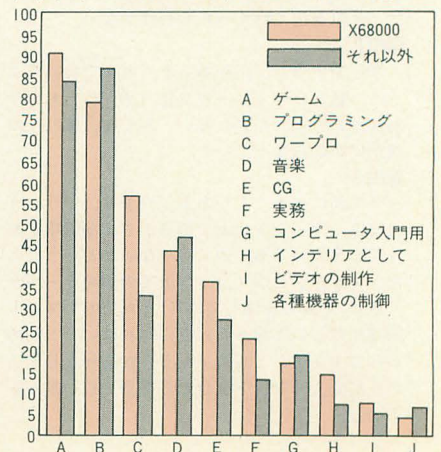
	全体	X68000	それ以外
ゲーム	87.0	90.1	83.7
プログラミング	82.5	78.8	86.5
ワープロ	44.9	56.3	32.6
音楽	44.5	43.0	46.1
CG	31.5	35.8	27.0
実務	17.8	22.5	12.8
コンピュータ入門用	17.5	16.6	18.4
インテリアとして	10.6	13.9	7.4
ビデオの制作	6.2	7.3	5.0
各種機器の制御	5.1	4.0	6.4

で，X68000ユーザーの90％がゲームに(も)使っているのに対し，X1やMZユーザーでは約84％であった。X68000以外のユーザーがゲームをしていないように見えるのはMZユーザーがあまりゲームしない（とはいっても，70％以上はあった）からである。で，この表には，他誌には見られないOh!Xならではの貴重なデータが含まれている。

それは，プログラミングをしているユーザーの割合だ！ BASICを理解できるのは100人にひとりとも1000人にひとりともいわれる軟弱な昨今。(善良な市民に)ゲームマシンだと思われているX68000で約79％，X68000以外の機種ユーザーでは86.5％もプログラミングユーザーがいるのだ。これはOh!Xの特徴かもしれないが，それを割り引いてみても，X68000ユーザーの半数以上は確実にプログラミングを楽しんでいる，と見ていいのである。

音楽，CGは何をかいわんや。ただ，X68000を買うとただでついてくるワープロについては意外と使用率が低いようである。

この手のアンケートでいつも謎なのが，“コンピュータ入門用として”使っていると答える人たちである。コンピュータ入門っ

グラフ1 X68000とX1，MZユーザーの違い1

てのは，何をするのだろうか。うーん。今度教えてください。私としては純粋にコンピュータ入門を考えた場合（つまり，コンピュータの動作の仕組みや，動きを知りたい！），最適のパソコンはMZ-700だと思う。複雑すぎないからね。X68000だって某売れ筋マシンに比べると（ある意味で）複雑でないから，入門には適しているけど。

あと，ビデオの制作なんかもいいけど，最近はカッコいいいい方をするとDTV（デスクトップビデオ）っていうのが流行ってるんだが，私もやってみたい。

各種機器の制御もいまひとつ謎。5％といっても，100人に5人である。どのように使っているのだろうか，具体的に聞いてみたい。とりあえず，私のX68000はいつもテレビの制御をしているが。あ，これは各種機器の制御とはいわないのか。

さらなる謎が，パソコン通信してる人がいない！ Oh!X読者にネットワーカーはいないのか！ あーあ。これの場合謎は簡単に解けるわけで，ただ単にアンケート用紙の選択肢に“パソコン通信”を付け忘れただけだったのだ！ 困ったもんだ。

2 皆さんの使う言語は!

Oh!Xの読者は実に多くがプログラミングしていることがわかった。私が思うに，どうしてプログラムを作るのかというと，決して実現したいプログラムがあるからではなく，プログラミングが楽しいから，ではないだろうか。あなたはどうですか？

というわけで表2である。とりあえず，開発に使っているっていう言語から見ていこう。

なし，っていうのはプログラム開発はしていない，ということだと思う。問題はない。

見ておきたいのは，さすがX1，MZユーザーのBASIC修得率が高い，ってこと。それから，アセンブラを使っている人間がこんなにいる，ってことだ。X68000ユーザーでさえ，Cよりアセンブラを使う人のほうが多い。うーん，ストロングブレンド。

ちなみに，X1，MZユーザーで3大言語

表2 言語について

総数	292	151	141

開発に使っている

ユーザー数	全体	X68000	X68000以外
BASIC	220	98	122
C	42	33	9
アセンブラ	92	35	57
SLANG	12	0	12
その他	30	7	23
なし	40	31	9

3大言語の割合

	全体	X68000	X68000以外
BASIC	75.34%	64.90%	86.52%
C	14.38%	21.85%	6.38%
アセンブラ	31.51%	23.18%	40.43%
その他	14.38%	4.64%	24.82%
なし	13.70%	20.53%	6.38%

一応理解できる

ユーザー数	全体	X68000	X68000以外
BASIC	192	98	94
C	71	43	28
アセンブラ	98	43	54
FORTRAN	25	13	12
Pascal	11	7	4
COBOL	9	2	7
LISP	6	2	4
SLANG	8	0	8
その他	23	5	18
なし	38	25	13

	全体	X68000	X68000以外
BASIC	65.75%	64.90%	66.67%
C	24.32%	28.48%	19.86%
アセンブラ	33.56%	28.48%	38.30%
その他	28.08%	19.21%	37.59%
なし	13.01%	16.56%	9.22%

よくわからないが関心あり

ユーザー数	全体	X68000	X68000以外
BASIC	17	16	1
C	161	83	78
アセンブラ	107	66	41
FORTH	19	7	12
LISP	14	4	10
Prolog	11	1	10
C++	7	4	3
その他	42	9	33
なし	44	24	20

	全体	X68000	X68000以外
BASIC	5.82%	10.60%	0.71%
C	55.14%	54.97%	55.32%
アセンブラ	36.64%	43.71%	29.08%
その他	31.85%	16.56%	48.23%
なし	15.07%	15.89%	14.18%

グラフ2-a プログラミング言語（みんなの気持ち）

グラフ2-b 機種別開発言語

グラフ2-c 機種別知ってる言語

グラフ2-d 機種別興味ある言語

(変ないい方だけど) 以外の言語を使う人が多いのは,S-OSのユーザーが多いことを示していると思われる。特に人気だったのがSLANGだ。

続いて,一応理解できるレベルの言語だ。Cの割合が上がる。

さらに,よくわからないけど関心がある言語となると,Cが断トツだ。みなさん,Cが流行しているのをしっかり気にしているようである。続いてアセンブラ,FORTH,LISP,PROLOG となる。BASICの割合が低いのは,すでにみんなマスターしちゃってるってことかもしれない。あるいは,もはやBASICの時代じゃないと思っているいるのか。

それにしても,みんな,凄い。X68000がマニアのマシンだといわれるのもわかる気がする。シャープのパソコンは,何かプログラムしたくなる魅力を持っているってことはいえるかもしれないし,ただOh!Xの読者にストロングユーザーが多いのかもしれないし,そういったユーザーしかアンケートを書かなかったのかもしれない。そうか,読者のうちわざわざ62円切手を貼る手間をかけてくれた人だけが対象だからなあ。プログラミングなんてしたかねえやい,てな人はアンケートに答えてくれなかったのかもしれない (ホントかよ)。

表2の結果を元にグラフ化したのがグラフ2-aから2-dである。とにかく,"なし"がいちばん少ないことに注目である。

表3　ベストライター

		得票数	割合
1	祝　一平	104	34.67%
2	荻窪　圭	25	8.33%
3	西川善司	19	6.33%
4	古村　聡	18	6.00%
5	高沢恭子	14	4.67%
6	泉　大介	12	4.00%
7	有田隆也	11	3.67%
7	中森　章	11	3.67%
9	村田敏幸	8	2.67%
10	金子俊一	6	2.00%
10	乗野雅彦	6	2.00%
	その他	32	10.67%
	なし	34	11.33%

グラフ3　人気ライターってやつ

3　Oh!X初のベストライター賞

Oh!Xではベストライターなるものを読者に聞いたことがなかった(と思う)。その理由を考えてみたが,私が思うに (今日はよくいろいろと"思う"なあ),祝一平先生(わざと先生と呼んでみたりする)が断トツでトップをとることがわかっていたからではないだろうか。

しかし,結果を見ればサルにでもわかるとおり,電脳倶楽部創刊以来Oh!Xでの執筆量が落ちているはずの祝一平氏が,2位を4倍以上突き放しての堂々1位であった。こうなると,祝氏がどうこうより2位が情けないのではないか。誰だ! 2位は?……私か。

あれま。てなわけで,2位以下は混戦である。該当者なしってのも多い。

3位と4位は1票差。最近台頭してきた若手コンビである。「ぱっくん」の西川善司とオムライス食べたいの (で) こと古村聡だ。その後,ベテラン陣が続く。注目は10位の乗野雅彦氏。登場回数は少ないが,最近は,"ジョイスティック端子につなぐ変な周辺機器"を作って楽しませてくれている。Oh!Xのスタッフになりたい人がいたら,このランキングを参考にするといいかもしれない (「それだけはやめてくれ」って声も聞こえるがきっと空耳だろう)。

4　このアンケートは公正か

と,いきなり問題のある集計が登場である。グラフ4と表4を見よ。Oh!Xを初めて買った,という人がほんの1%で,ほとんど毎月買っているという固定読者が90%以上ではないか。

表4　Oh!Xを……

	人数	割合
初めて知った	0	0%
初めて買った	3	1.00%
ときどき買う	16	5.33%
ほとんど毎月	280	93.33%
無効	1	0.33%
TOTAL	300	

いくらOh!Xがフツーでないとはいえ,この結果は異常である。諮問委員会で検討した結果,「初めてOh!Xを買った人やときどき買う人にはアンケート内容がヘビー過ぎた」のではないかという結論に達した。つまり,初めて買った人もいるはずなのに,彼らはアンケートを出せなかったのだ。そうとでも思わないと,悲しすぎる。だってねえ,普通の本屋さんに普通に置いてある雑誌なのに,初めての読者が1%しかいないなんて。もしや,アンケートのついた2月号の特集「画像圧縮へのアプローチ」の内容が,初めて買おうという人をビビらせたのだろうか。ちなみに毎月の愛読者ハガキを読むと「こんどX68000を買ったので初めてこの本を買いました。こんな僕にでもわかるような簡単な記事を載せてください」とかいうメッセージが少なくとも数%はある。

ともかく,今回のアンケート結果すべてにおいて,ほとんど毎月買っている読者の答えが中心だ,と思ってください。「俺はX68000を買ったけど,ゲームしかしてないぞ」と思うあなたも次回はアンケートに答えてちょ。え,そのころにはプログラムガシガシかもしれないって?

そうそう,今月はディスク付きてなもんで初めてOh!Xを買った,なんて人も多いのではないだろうか。そんな人にごあいさつ。「どーも,荻窪です。Oh!Xの読者はこんなヤツばっかです。よろしく」。

5　息抜きにAV機器の所有率でもどうぞ

300人中CDプレーヤーを持ってる人は?

グラフ4　Oh!Xを……

ではレーザーディスクはどうだ？　そんなら，19インチ以上の大型テレビはどうだ？大型テレビといえば，売れているそうですね，三菱のCZ。うーん，液晶ビジョンでポピュラスやダンジョンマスターしたい。ねじ式も不気味でいいかもしんない。

そんでもって，VTRだ。79%の人が持っている(CDプレーヤーを持っている人より多い)。そのうち35%が2台以上のVTRを持っているわけで，そういう時代なのであった。くわばらくわばら。

6　X68000に積まれたメインメモリ

さあ，ここからあとはX68000シリーズ数字のマジックってやつである。

まず，あんたのX68000にはRAMがどんだけ乗ってるの？　が表6とグラフ6。

1MBより2MBのほうが多いのは，EXPERTが標準で2MB積んでいるせいかと思いきや，実は元祖のユーザーでは半分弱，ACEでは半分以上のユーザーがRAMを増設しているのだ。というわけで，蛇足とは思うが，Oh!X読者（正しくはアンケート回答者）のX68000に積まれたRAMは平均1.65MBであった。

4MBユーザーがまだまだ少ないが，アイ・オー・データ機器から安くRAMボードが出たからこれからは増えるだろう。

7　X68000につながれた磁性面

とりあえず，みんな2HDの2ドライブはある。片方を1MBとして(本当は1.2MBだけど)，最低2MBは一度に使える磁性面を持っているわけだ。

それに表7のハードディスクが加わる。グラフは7番。

グラフを見るとわかるが，なんだかんだいって，まだハードディスクのないユーザーのほうが（若干）多い。もちろん，内蔵型のモデルも数えての話だ。で，20MBハードディスクがいちばん多いが，実のところ，ACE-HDの内蔵しているハードディスクが20MBであるせいだったりする。元祖に20MBをつないでいる人は元祖のサンプル数53人のうち2人しかいないのだ。40MBをつないでいるユーザーは8人いるのに。

というわけで，時代は40MBどころか，SCSIインタフェイスの80MBへ移り変わろうとしている。40MBHDと80MBHDの価格差は5万円以下まで縮まっているので，80MBも狙い目かもしれない。

蛇足だとわかってはいるが，FDを含めたX68000 1台当たりの平均磁性面容量を算出してみた。結果は，約16MB。平均メインメモリの約10倍って，ところ。その昔，X1 turboのメインメモリは64KBで2Dのディスク2ドライブで640KBだった。当時も今もメインメモリ×10が磁性面容量だったりして，それが健全な姿なのだろうか（根拠なし)。

8　X68000につながったその他

表8とグラフ8が周辺機器と主な拡張ボードである。とりあえず，70%以上の人がプリンタを持っていて，80%以上の人がジョイスティックを持っているという，実にX68000らしい集計結果となった。さらにサイバースティックが約22%となっては，もはやなにもいうまい。

パソコンの用途のアンケート欄にパソコン通信を入れ忘れた，と書いたが，この表のモデム所有者を見れば，予想はつくだろう。サイバースティックより普及率の低いモデムってわけだ。

なお，MIDIインタフェイスボードが思ったより普及していて驚いた。MIDIインタフェイスボードがある人は100%がMIDIコネクタ付きの電気楽器を持っているわけだからね。

9　ソフトウェア編

ソフトウェアについては，主なPRO-68KシリーズとXC，それからゲームソフトについて調べた。表9とグラフ9である。見るとき気をつけるのはHuman2.0。元祖の

表5　AV機器

	持ってる	持ってない
CD	214	86
LD	35	265
TV (≧19inch)	72	228
VTR	237	63
(VTR1)	154	
(VTR2)	71	
(VTR3≦)	12	

グラフ5　AV機器の所有率

表6　X68000ユーザーの持っているメインメモリ

1MBの人	62(人)	41.06%
2MBの人	85(人)	56.29%
4MBの人	3(人)	1.99%
8MBの人	1(人)	0.66%
合計	151(人)	

ひとり当たりのメモリ	
平均	1.65562914(MB)

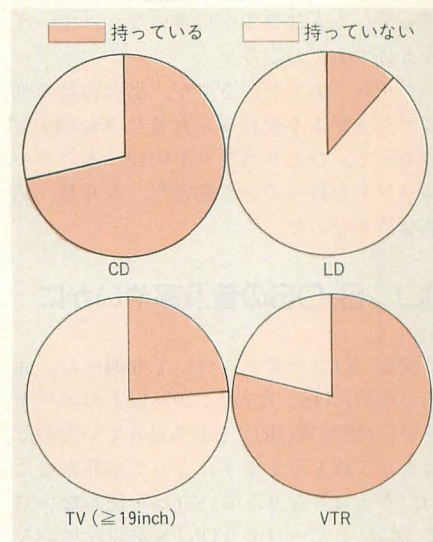
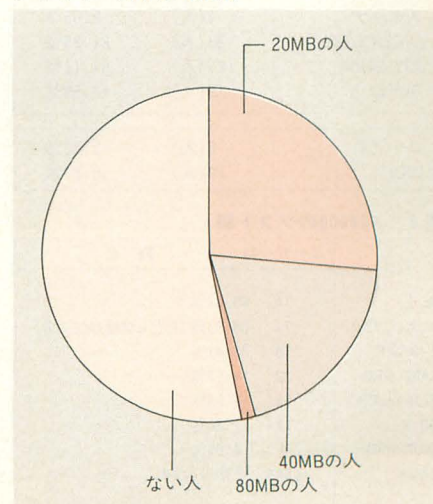



表7　X68000ユーザーの持っているHD

20MBの人	40(人)	26.49%
40MBの人	29(人)	19.21%
80MBの人	2(人)	1.32%
ない人	80(人)	52.98%
合計	151(人)	

ひとり当たりの磁性面(FD2MB含む)	
平均	16.0397351(MB)

グラフ6　MAIN MEMORY

グラフ7　HARD DISK

ユーザーとACEのユーザーだけがHuman 1.0を持っているわけで，表9の割合は元祖とACEだけを対象にした数字である。しかし，PROとEXPERTユーザーもHuman2.0を使っているわけで，グラフのほうはとにかくHuman2.0を使っている人の割合である。わざとなので，混乱しないように。

こうしてみると，XCはほぼ半数のユーザーが持っているわけである。その割に，表2でわかるようにCを使っている人はそんなにいないわけで，“XCを買った”and“Cを使っていない”って人はきっと，BASICコンパイラとして使っているな，あるいは本棚を派手に飾っているな。Z'sSTAFFを持っている人はさすがに多いな，っていうのも押さえておこう。

ゲームであるが，さすが，出荷台数の割にソフトがよく売れると有名なX68000だけあって，ひとり当たり平均12.9本のゲームソフトを持っている勘定だ。みんな，結構金持ちなのだ。

10 S-OSの普及率やいかに

MZ，X1ユーザーについても調べた。面白いのが，OS。なんと，70%以上のユーザーがS-OS“SWORD”を打ち込んでいたのだ。しかしこれも考えようによっては非常にこわい考えにもなりうる。S-OSを使わないMZ，X1ユーザーはもうOh!Xを読んでいないんじゃないか……。

しばし，沈黙。

復活。で，プリンタとジョイスティックとモデムについても調べたのだが，サンプル数が少なくて，このパーセンテージは今ひとつ信頼性に欠けるかもしれない。でも，このジョイスティックの普及率と，MZのモデムの普及率はなかなか。“通信パソコンMZ-2500”の面目躍如といっていいだろう。

11 過去5年間における勢力推移

最後に，2月号のアンケートには関係ないけれど，Oh!Xでは5年前から“言わせてくれなくちゃだワ”，通称“ちゃだワ”という読者特集を開催している。文福ちゃだワ，ってやつだ。

毎回，愛読者カードを1000枚ほど無作為抽出し，読者の所有機種集計を掲載しているのだが，これがなかなか面白い。

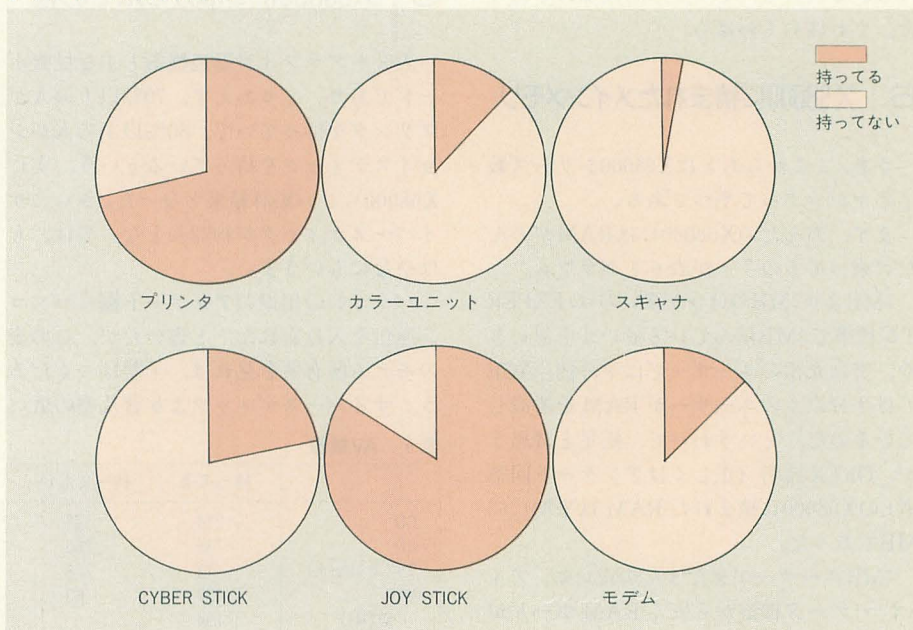
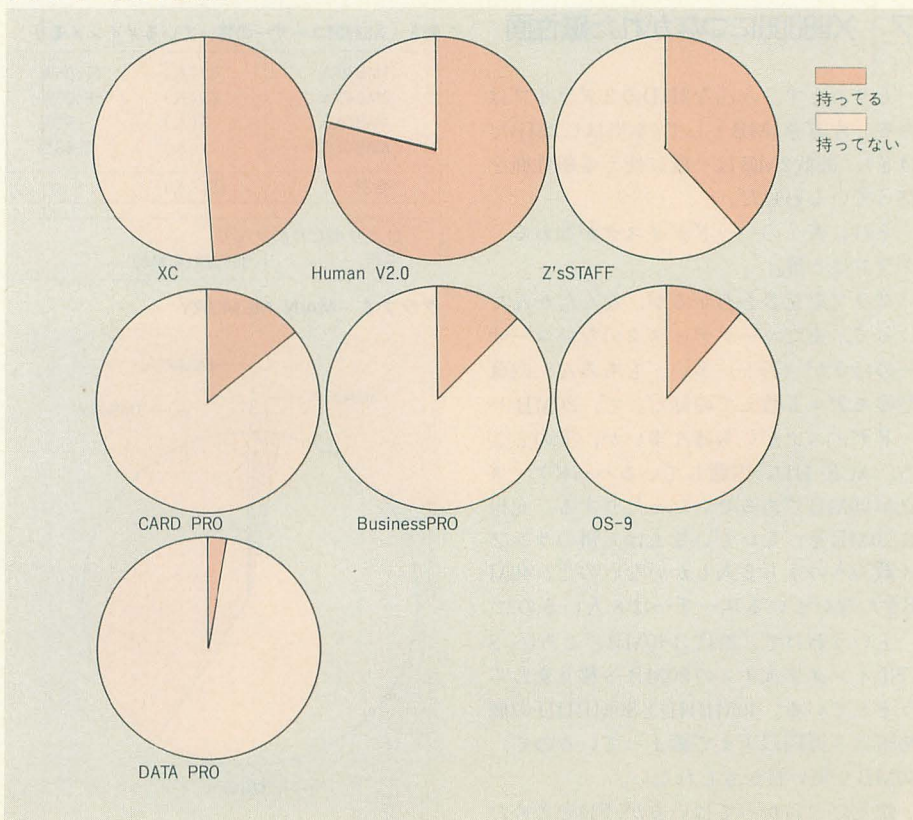
小手調べとして，その結果を表11にまとめた。'86年だけはサンプル数が違うため，

表8　X68000ユーザーの周辺機器所有

プリンタ	108(人)	71.52%
カラーユニット	18(人)	11.92%
スキャナ	4(人)	2.65%
CYBER STICK	33(人)	21.85%
JOY STICK	127(人)	84.11%
モデム	19(人)	12.58%
コ・プロ	8(人)	5.30%
MIDI	14(人)	9.27%

表9　X68000のソフト編

	ユーザー数	割合
XC	74	49.01%
Human V2.0	71	68.93%(ただし元祖/ACEのみ)
Z'sSTAFF	58	38.41%
CARD PRO	22	14.57%
BusinessPRO	19	12.58%
OS-9	17	11.26%
DATA PRO	4	2.65%
GAME	1948	平均 12.9本

グラフ8　HARDWARE

グラフ9　SOFTWARE

1000枚当たりに換算してある。これはまさに,ハガキ1000枚のうち,読者はどのマシンを使っていたか,ってやつだ。ほら,そこのMZ-X1ユーザーのあなた,目をそらさないように。未来はすべての人に平等にやってくるのです。

1986年。X1,MZ-700,MZ-2000の三つ巴,3すくみ3種の神器トリプルファイターザンボット3などというくらいひしめきあった時代の最後の年である。この前の年のデータが残っていたなら,もっと凄かったに違いない。すでにX1turboが迫ってきている。その頃私が何を持っていたかというと,不可能はないといわしめたMZ-700であった。

1987年。MZシリーズ最後の砦,起死回生のスーパーMZ(MZ-2500)が登場する。発売が'85年の10月だから,1年半でここまで伸びたわけだ。ただし,MZ-2000のユーザーががくんと減っているところを見ると,2500には2000からの買い換えユーザーがかなりいたと思われる。ここで,MZは700/1500と2500の時代になるわけだが,いかんせん,X1turboは強かった。地上最強の8ビットパソコンスーパーMZもかなわなかったわけだ(ちなみに,私もスーパーMZを買ったひとりである)。この年の年末,Oh!MZはOh!Xとなる。それ以来,Oh!Xの創刊記念月が2つ(Oh!MZ創刊の6月と,Oh!X創刊の12月)になる。

1988年。'87年の春にいきなり登場したX68000が売れる。驚異的な数字。ちなみに私もX68000を買ったひとりである。

そんなこんなで2つのグラフを見てほしい。MZ,Xを互換性を考えていくつかに分類し,勢力推移を見たものである。たった5年間にどれだけ動いたことか。明日は我が身。

諸行無常といえばNTT株であるが,グラフ"諸行無常"は各機種が全体の何%を占めているか,を語るものである。見てのとおり,X68000が増えているぶん,減っているのがMZである。とうとう十数%になってしまった。来年はどうなるのか。うーん。また,グラフ"花のいろはうつりにけりな"はもっと露骨である。よくわからないが我が身よにふるながめせしまに,である。盛者必衰である。

とにかく目立つのがX68000。衰えを知らぬ伸び。この勢いがいつまで続くか見ものだな。まだ全盛期のX1+X1turboには及ばないが,かなりのものだ。ひと言添えておこう。X68000が増えてX1やMZユーザーが減ったというよりも,X1ユーザーが(相変わらず多いけれども)Oh!Xを読まなくなったという見方もできないことはない。数字を読むのは難しいのだ。

'88年がX1turbo最盛期だった。MZ-2500が発売直後の伸びから凋落したが,ここ2年は安定している。いいマシンであることは確かなのだから,手放さないユーザーは手放さないだろう。私は手放しちゃったけど。

さて,ここでアブないグラフの登場である。グラフ3は"いたづらに"である。え?"花のいろはうつりにけりな"とどう違うのかって? それはあなた,91/03のデータがあるではないですか。内緒の話だが,私は予知能力者だったのだ。ふっふふ。実は,1995年までに東京に大地震が起きるので,首都圏のみなさんは気をつけるように(こらこら)。だいたいさあ,地震でこけちゃったらさあ,高層億ションなんて,ただの猫の額しか残んないんだぜ。ディズニーランドなんて水かぶっちまうんだぜ。土地の値段なんてそんなもんさ。

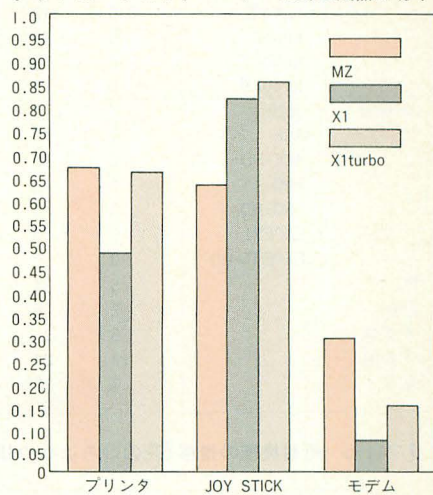
あっと,大ボラはやめて話を戻そう。

表10　8ビットユーザーのソフト&ハード所有率

	S-OS	CP/M
MZ	74.07%	33.33%
X1	81.48%	25.93%
X1turbo	80.46%	41.38%

	プリンタ	JOY STICK	モデム
MZ	66.67%	62.96%	29.63%
X1	48.15%	81.48%	7.14%
X1turbo	65.52%	85.06%	14.94%

この91/03のデータは,KamikazeのTREND関数で出したものだ。この関数は入力されたX,Y座標のデータを元に,あるX座標の値を入れると,そのY座標の値を予測するってなもんだ。で,実のところ回帰直線のパラメータを計算しているだけだったりするから,グラフを作るときにはうまく直線になるような座標を与えてやらないといけないわけで,山形のX1turboは88/03〜90/03の,MZ-2500は87/04〜90/03の,X68000は88/03〜90/03のデータで予測している。もっとちゃんと予測できるんだろうけど,とりあえずこんなもんで濁ったお茶。ちゃんと来年にはX68000は1000人中850人が持っていたりするんだろうか(かなりマユツバだけど,ありえない話ではないな)。

グラフ10　8ビットユーザーの周辺機器所有率

創刊8周年に寄せて

8周年だーっ! ということでやってきましたOh!X8回目の誕生日です(本当はOh!MZ時代も含めてだけど)。さあっ,この次は9周年でその次は10周年になるわけだ。で,その次は11周年,その次は12周年……えっ,一生そうしてろって!? こいつは失礼しました。だって,この前よーく考えてみたら私も結構古いスタッフなんだよなー,なにせ初登場はあれだったから,えーと……さて,ここで問題です。私の初登場は何年,何月号の何というゲームのレビューだったでしょう? 絶対ひとりも当たらないだろうけど(絶対あの辺と間違えるもんね)。

おっとっと。べつにそんなことを書こうと思っていたわけではないんだった。これだけ長い間だったからOh!Xにもいろんなことがあったわけです。雑誌名は変わったし,新機種も出たし,スタッフも読者もやめる人がいれば新しい人が入ったりも当然のようにあったわけです。

でも,8年たっても(もっとも私だってそのあいだずーっとライターやってたわけじゃないけど)変わらないものがある! そう,それは「Oh!XはOh!Xであり続けている」ことなのです。

昔はOh!MZはドラゴンであるなんてこともいっていた時代もありました。いまはドラゴンな記事(というか骨のある,っていえばいいんだろうか)もあり,善ちゃんや私のようなスライム級の記事もありで,自分で言うのもなんだけど,パソコンを買ったばかりの初心者からCもマシン語もI/Oたたいてガシガシのパイパー上級者まで皆様に親しまれ愛されるOh!Xであったりするのです(うー,自分で言うと恥ずかしいもんだなあ)。

でも,ですね,いままでのOh!Xをよーく,見てください。決して「これ打つと遊べるよー」という単なる紹介ではなくて「これはこのようにするとこうできるのであった。あとは各自の自由課題である」とか「さあ,みんなもがんばってプログラム作ってね」とか読者の挑戦をお誘いしているものなんですよね。やっぱり読者にもっと上を狙ってもらおう,そのためにはわしらがすすんでドラゴンでもスライムでもなってやろうじゃないの,というこの心意気であったりするのです。

さて,次の9周年記念のとき,皆さんはどのくらいレベルアップしてるでしょう? Oh!Xはいつまでもドラゴンで待ってます。　(で)

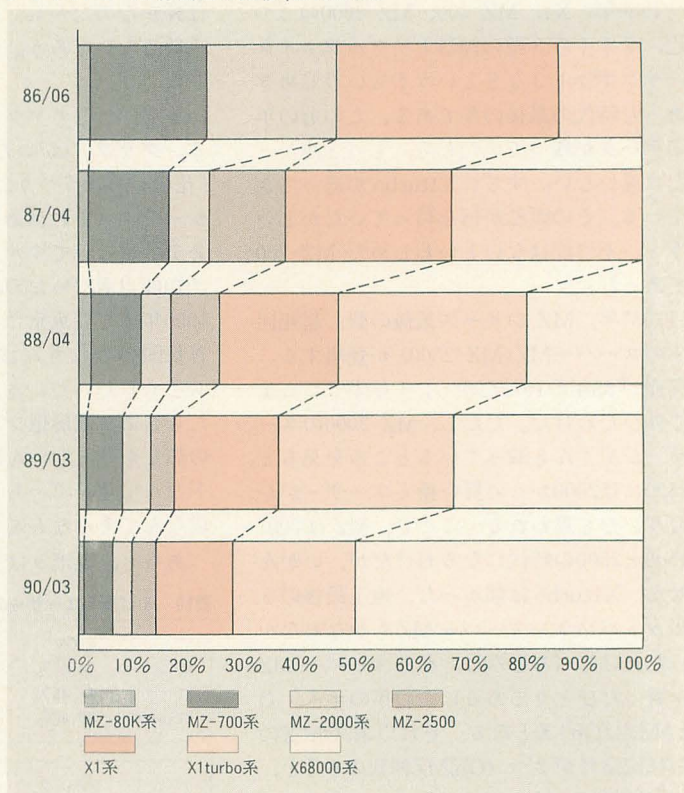
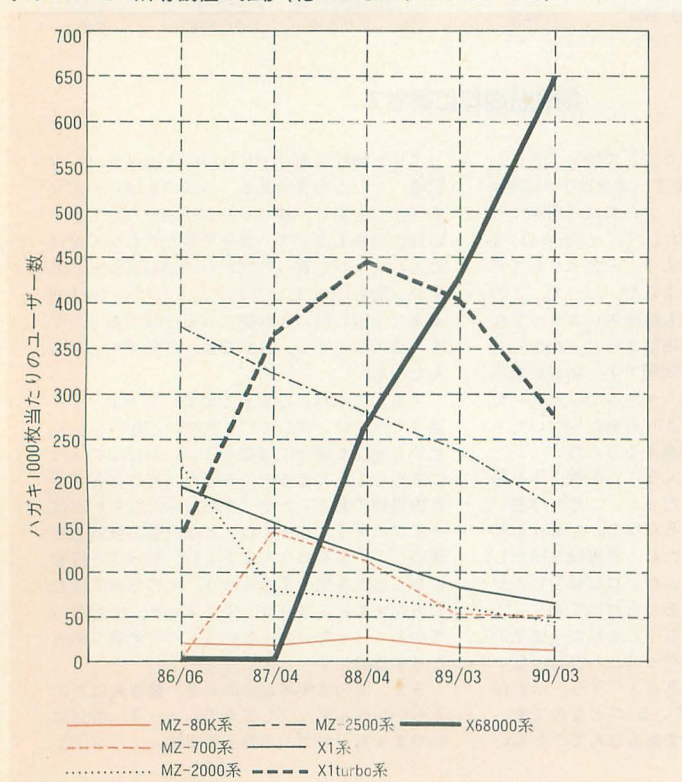
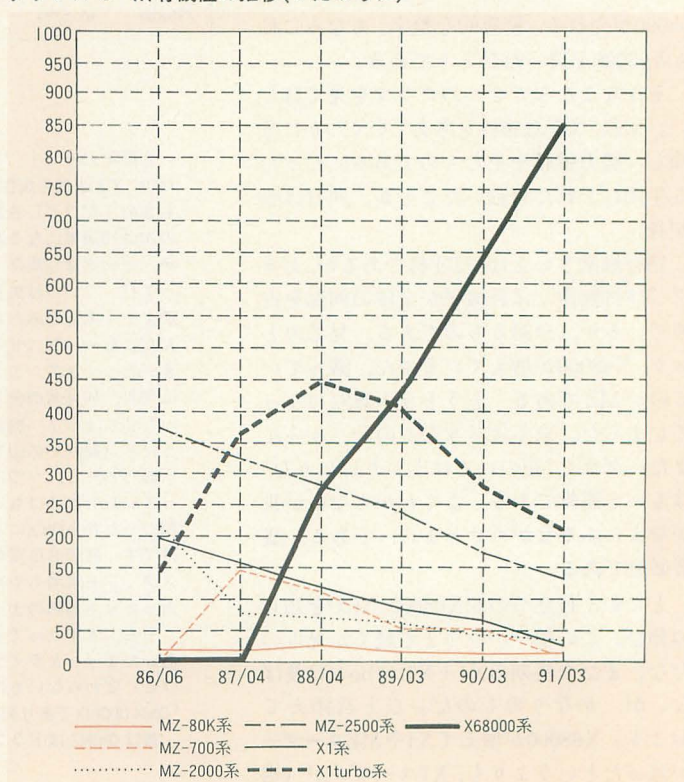
というわけで，終わり。アシスタントのKamikazeくんもなかなか働いてくれて，ごくろうさんである。極度に本格的なビジネスに使うには問題があるけれど，今回のようにちょっとした作業に使うにはなかなか楽しいものだ。

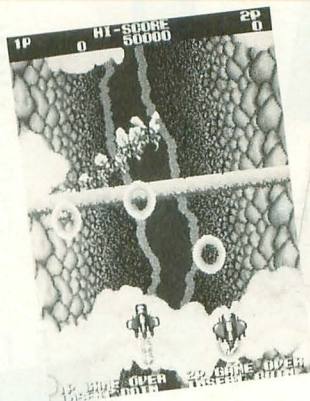
今回はこんな結果が出たが，重要なことはこういったアンケート結果に惑わされず，自分がいちばん面白いと思うジャンルで面白いと思う方法でパソコンと遊ぶことである。メディアだって道具だっていいじゃないか。次回はもっとトッピーな使い方のユーザーを望むぞ。

外はもう汗ばむ妖気。じゃなかった陽気である。ごきげんよう。

表11　所有機種

		86/06	87/04	88/04	89/03	90/03
MZ	80K/C/1200	20	18	27	16	13
	700/1500	196	155	118	84	66
	80B	27	12	11	7	3
	2000/2200	193	67	59	57	39
	2500	0	145	113	53	52
X1	マニア	146	91	73	56	27
	C/Cs/Ck	173	94	65	72	46
	D	54	32	31	25	16
	F	0	67	60	41	32
	G	0	38	38	42	45
	twin	0	0	13	1	7
	turbo	140	163	166	146	90
	turboII	0	151	112	101	53
	turboIII	0	12	55	27	20
	turboZ	0	39	86	92	58
	turboZII	0	0	26	33	43
	turboZIII	0	0	0	4	13
X68000	元祖	0	0	269	220	152
	ACE	0	0	0	91	91
	ACE-HD	0	0	0	118	124
	PRO	0	0	0	0	101
	PRO-HD	0	0	0	0	32
	EXPERT	0	0	0	0	88
	EXPERT-HD	0	0	0	0	56
PC		5	34	60	57	25
FM		2	16	13	5	5
ポケコン		6	31	25	55	43
その他		18	27	47	71	25
なし		7	20	19	19	22

グラフ11-a　所有機種の推移（諸行無常）

グラフ11-b　所有機種の推移（花のいろはうつりにけりな）

グラフ11-c　所有機種の推移（いたづらに）

創刊8周年記念 愛読者特大プレゼント

早いもので，Oh!MZとして世に出てから(途中Oh!Xと改名はしたものの）本誌はなんと！　創刊８周年を迎えることとなりました。いや～，めでたいめでたい。これもひとえに読者の皆さんのお力添えがあったからこそです。う～ん，ありがたや！　今後も皆さんの期待に応えるよう，編集部一同，頑張っていきたいと思っていますので，いままで以上によろしくお願いします。
さて，創刊８周年を祝してOh!X編集部からは，読者の皆さんに感謝の気持ちを込めて，プレゼントがわりに付録にディスクを付けたわけですが，いつもお世話になっている各ソフトハウスさんからも，たくさんのプレゼントをいただきました。さすがに８周年ともなると，プレゼントのラインアップも豪華なもんです。見てください。この充実したプレゼントの内容を！　うれしいじゃあないですか。これだけいろいろなプレゼントがあると，どれにしようか迷ってしまいそう(そんなこともないか！)。
さあ，応募方法をよく読んで(いつもと同じだけど……)，奮ってご応募ください。皆さんのご応募をお待ちしております。

システムサコム　☎03(635)7609

1

A)ジェミニウイング
X68000用　5"2HD版2枚組　8,800円　1名

B)闇の血族
X68000用　5"2HD版　8,800円　1名

システムサコムからはこれから発売する予定の新作２本をプレゼント。どちらも見逃せないソフトだ。

ツァイト　☎03(299)0461

2

ねじ式
X68000用
5"2HD版4枚組
12,800円　2名

つげ義春の世界をパソコンゲームで見事に描いたアドベンチャー。人気のほどはGAME OF THE YEARでご存じですよね。

ロゴスシステム
☎075(812)6383

3

The File Professor
X68000用　5"2HD版　28,000円　2名

ディスクの管理をしやすくしてくれるファイルマネージングソフト。これひとつあると結構便利だ。

アンス・コンサルタンツ
☎092(522)6347

4

サイクロンEXPRESSα
X68000用　5"2HD版　98,000円　5名

CGツール，サイクロンの最新バージョン。これはモニタプレゼントなので，心して応募してください。

アートディンク
☎0474(77)7541

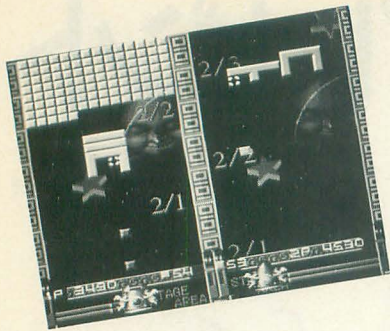
5

A)FAR SIDE MOON
X68000用　5"2HD版3枚組　9,500円　2名

B)A列車で行こうII
X68000用　5"2HD版4枚組　12,800円　2名

C)大海令
X68000用　5"2HD版8枚組　13,800円　2名

D)南海の死闘
X68000用　5"2HD版4枚組　9,500円　2名

なんと一挙に４作品もプレゼントしてくれるとは，太っ腹（B，C，Dは限定版！)。必ず希望ソフト（5-A，5-B，……）を明記してくださいね。

コナミ
☎03(262)9110

6 クォース

X68000用　5"2HD版　6,800円　3名

落ちてくるブロックに弾を打ち込み四角形にして消していくシューティングパズルゲーム。ゲーセン版の移植だ。

ビクター音楽産業
☎03(423)7901

7 ジャック・ニクラウス・テレホンカード

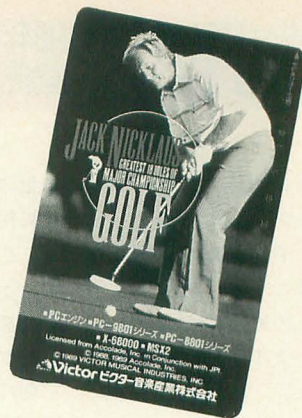
5名

ビクター音産からは，ジャック・ニクラウス・チャンピオンシップ・ゴルフのテレホンカードをプレゼント。

アルシスソフトウェア　☎0956(22)3881

8 スタークルーザー

X68000用　5"2HD版2枚組
8,800円　2名
X1turbo用　5"2D版2枚組
7,800円　2名

いつも斬新なゲームを出してくれるアルシスからは，あの名作スタークルーザーをプレゼント。機種名を必ず明記してくださいね。

エス・ピー・エス　☎0425(45)5777

9 キューブランナー

X68000用　5"2HD版
7,800円　3名

ブロックを操作して転がる玉を誘導し，全部のレールを通過させるというアクションパズルゲーム。

ヘルツ
☎03(371)3012

10 レナム

X68000用　5"2HD版6枚組
9,800円　5名

アドベンチャーの要素を含むロールプレイングゲーム。グラフィックもなかなか美しい。

コムパック　☎03(375)3401

11

A)ガンマ・プラネット
X68000用　5"2HD版　6,800円　3名
B)グランディフロラム
X68000用　5"2HD版　7,800円　3名
C)Simple-CAD X68K
X68000用　5"2HD版　19,800円　2名

コムパックからは，ゲームソフト2本とCAD用ソフト1本をプレゼントしていただきました。これも希望ソフトを明記してください。

ハドソン　☎011(841)4622

12 上海II

X68000用　5"2HD版
6,800円　2名

大人から子供まで楽しめるパズルゲーム。全面解いたときのグラフィックったらもう……。

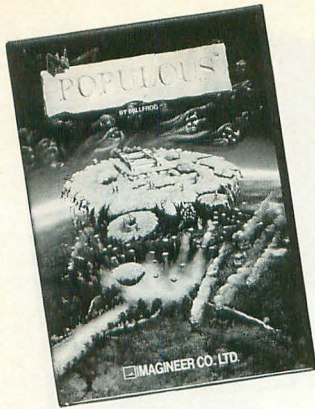
イマジニア ☎03(343)8911

13

ポピュラス

X68000用 5"2HD版

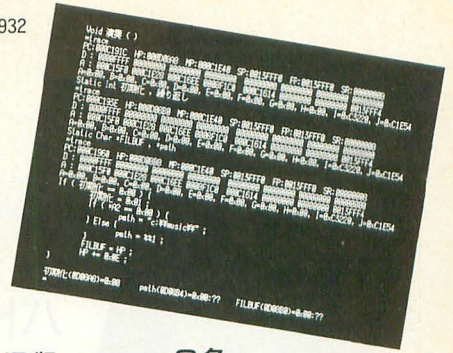
9,800円 3名

もう説明の必要もないくらい人気のあるこのゲーム。これは競争率が高そうだなあ。

サザンエンタープライズ ☎03(787)3932

14

プログラムオペレーティングシステム

X68000用 5"2HD版 10,000円 2名

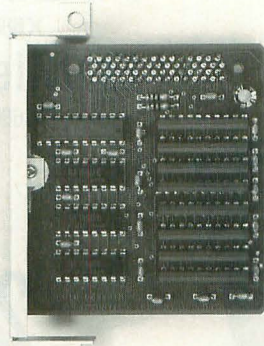
バッチ処理のようなプログラミングにも，C言語のようなプログラミングにも対応できるインタプリタです。

アイ・オー・データ機器

☎0762(21)4812

15

PIO-6BE1-A

25,000円 1名

X68000用のメモリ拡張ボード。機能的には純正品とほぼ同じだが，低価格を実現したものとして評価されている製品だ。

OH! BUSINESS ☎075(502)2972

17

G68KII

X68000用 5"2HD版 22,000円 3名

グラフィックツール68Kのバージョンアップ版。手軽なグラフィックツールだ。

ボーステック

☎03(708)4711

16

銀河英雄伝説＋set

X68000用 5"2HD版4枚組 11,600円 3名

人気のシミュレーションゲーム。今回はシナリオ集とセットになったものをプレゼント。

ワールドインアオヤマ ☎03(985)9011

19

A)オリジナルコーヒーカップ

3名

B)ツインビー

X68000用 5"2HD版 7,800円 2名

なんと，ショップさんからも今回はプレゼントがあるんですねぇ。コーヒーカップとツインビーです。

日コン連企画 ☎06(644)6901

18

A)D-RETURN

X68000用 5"2HD版2枚組 5,980円 2名

B)ずるかまし

X68000用 5"2HD版2枚組 5,980円 2名

廉価版ソフトの販売で有名な日コン連企画。その日コン連企画からは，D-RETURNと，翻訳ヘルパーずるかましをプレゼント。

パック・イン・ビデオ ☎03(5565)8732

20 バトルチェス

X68000用 5"2HD版2枚組

9,800円 1名

パック・イン・ビデオからは，駒の動きが楽しいバトルチェスをプレゼント。環境ソフトとしても楽しめる。

データウエスト ☎06(968)1236

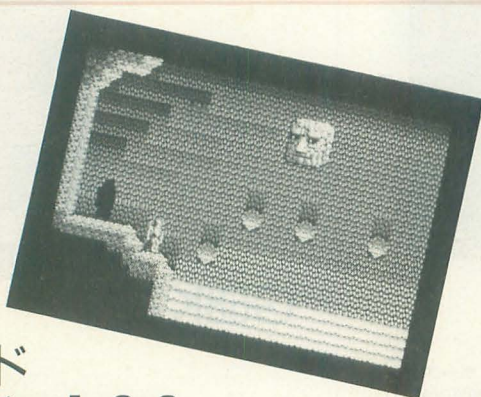
21 A)Zerø

X68000用 5"2HD版 8,800円 2名

B)Misty3

X68000用 5"2HD版 5,000円 2名

いつも面白いソフトを出してくれるデータウエストからは，ZerøとMisty3をプレゼントしてくれるぞ。

ブラザー工業
☎052(824)2943

22 セレクテッドソーサリアン 1,2,3

X1turbo用 5"2D版 2,900円 各3名

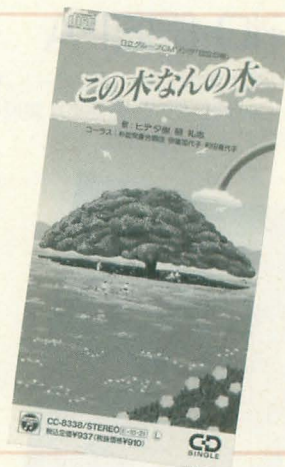
いまやX1ユーザーの救い主となっているソーサリアンシリーズ。ブラザー工業さんからセレクテッドソーサリアンをプレゼントしてくれる。

24 「この木なんの木」のCD 3名

今回のLIVE inで紹介した「この木なんの木」のCDをプレゼント。なんと日立では，お昼休みの終わりにこの曲がかかるそうな。

コナミ ☎03(262)9110

23 ウインドブレーカー 5名

コナミはソフトのほかにウインドブレーカーまでもプレゼントしてくれました。これは5名の方に。

プレゼントの応募方法

とじ込みのアンケートはがき(ただし今月号のもの)の該当項目をすべてご記入のうえ，希望するプレゼント番号をはがき右下のスペースにひとつ記入してお申し込みください。締め切りは1990年6月18日の到着分までとします。当選の発表は1990年8月号で行います。

4月号プレゼント当選者

1セレクテッドソーサリアンI(東京都)愛沢太郎(千葉県)境武志(和歌山県)山野光啓 2ファーストクィーン(東京都)阿部隆仁(大阪府)谷口薫(新潟県)保科康広 3スーパーハングオン(神奈川県)海老原寛孝(香川県)福井利夫(鹿児島県)鈴山修 4ベルリンの壁(山形県)玉木和威(東京都)深井明彦(京都府)小原良宣 5ミュージックCD(宮城県)髙嶋涼子(千葉県)五十嵐豊

以上の方々が当選されました。おめでとうございます。商品は順次発送いたしますが，入荷状況などにより遅れる場合もあります。また，公正取引委員会の告示により，このプレゼントに当選された方は，この号の他の懸賞には当選できない場合がありますのでご了承ください。(敬称略)

(価格はすべて消費税別です)

THE SENTINEL

●5周年だS-OS！

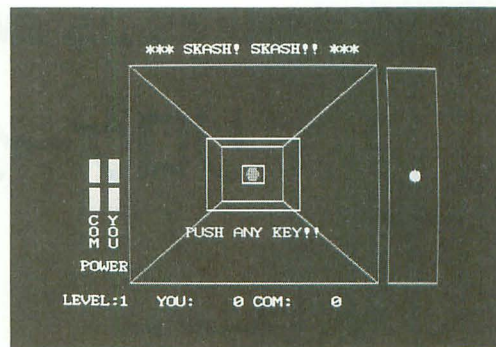
早いものでS-OS“MACE”がスタートして5年の月日がたちました。機種を超えての共通化という，パソコン雑誌初の試みに参加してくださった多くの皆さんの熱意とご厚意に改めて感謝したいと思います。MZ/X1ユーザーを始めとして，PC-8801シリーズ，FM-7シリーズなどのユーザーが，同じコンピュータ者としてともに手を取り合い同じ土俵で話しあえる環境，それがS-OSです。機種を超えての対話が今日まで続いてきたことは，ゆるぎない読者パワーのなによりの証だといえるでしょう。

最初はごく標準的な機能だけを集めたサブルーチン集のようなものだった“MACE”は，数多くの読者の創意と情熱によって成長し，“SWORD”へと進歩を遂げました。さらに“SWORD”にも数々の手が加えられ便利になったと同時に，そのアプリケーションのラインアップも充実の一途をたどり，来月からはかねてからの懸案であったリロケータブルファイルをサポートするところまで成長しました。成長するシステムS-OSは，ユーザーの皆さんの情熱のある限り立ち止まることはありません。投稿，ご意見，なんでも結構です。S-OSの試みに皆さんもどんどん参加してください。

第93部
リロケータブルフォーマットの取り決め
第94部
STACK用ゲーム SQUASH!
第95部
X68000対応 S-OS“SWORD”
特別付録
PC-286対応 S-OS“SWORD”

●リロケータブルアセンブラ

待ち望まれていたS-OS用のリロケータブルアセンブラがついに登場です。これまでのZEDAやREDAでは，アセンブルする時点で指定したアドレスでしか作成したプログラムを実行できないアブソリュートアセンブラでした。

リロケータブルアセンブラは，まだアドレスの確定していない中間的なオブジェクトを生成するアセンブラです。実際に実行可能なプログラムを作るには，さらにリンカでアドレスを決定しなければなりません。リンカは複数の中間的なオブジェクトをひとつのプログラムにまとめるという使い方もできます(こっちが本業)。したがって，大きなプログラムを複数のソースに分けて作る場合，

エラーなく無事にアセンブルが終了したソースは再びアセンブルし直す必要がないというメリットが生まれるのです。よく使うサブルーチンを中間的なオブジェクトとして用意しておけば，メインルーチンを作るだけでプログラムが開発できることにもなります。あとはライブラリアンがあれば，開発環境は完璧ですね。期待したいところです。

●STACKサンプルプログラム“SQUASH！”

先月掲載したSTACKはいかがでしょうか。このところ馴染みのなかったスタック指向の言語ということで，使い方がよくわからなかった方もいらっしゃったのではないかと思います。そこで今月はSTACK用のサンプルプログラムをお届けします。STACKの考え方の基本は，スタックを利用したプログラミングにあります。少々慣れが必要な考え方ではありますが，基本は単純なものです。ぜひ入力してトライしてみてください。

●X68000 & PC-286用“SWORD”

お待たせしました。Xシリーズの中では唯一S-OSのなかったX68000用のS-OS“SWORD”の登場です。高性能で使いやすいMC68000といえど，やはりほかのCPUのエミュレーションは苦手のようで，スピードは1MHzのZ80くらいということです。とはいえ，これでX68000上でもZ80のプログラムが作れる環境が揃ったわけです。

ほかのCPUのエミュレーションでもっとも大変なのはフラグを変化させなければならないということです。これがなければ該当命令を作るのはそれほど大変な作業ではなくスピードも出るものなのですが。

PC-286用“SWORD”はスペースの都合でオブジェクトのみの掲載となってしまいましたが，PC-286シリーズ用の“SWORD”が完成しました。もちろん，互換機でも動作が確認されていますので，安心してご使用ください。

さてこれで，S-OSはメジャーな16ビット機までその範疇に収めました。8ビット機で残るものもあと少し。みんなでS-OSの世界制覇を目指しましょう。

全機種共通S-OS"SWORD"不用

リロケータブルフォーマットの取り決め

Ishigami Tatuya
石上 達也

全機種共通システムも5周年を迎えました。そして，今回，S-OSの新たな可能性を開くリロケータブルファイルのフォーマットを取り決めました。リロケータブルアセンブラ/リンカの発表は来月行いますのでご期待ください。

あれから4年

今月をもって，このTHE SENTINELのページも5周年目を迎えることができました。いま，手元に編集部からお借りした創刊3周年記念号のOh!MZがあります。当時はX68000などまだなく，IIでもIIIでもない元祖X1turboが発売されたばかりでした。広告を見てもゲームなどはテープ版が主体で，フロッピーディスクはまだまだ高価なものでした（いまでも安いとはいえないが）。

"SWORD"の前身にあたる"MACE"もフロッピーディスクへの対応はなされていませんでした。X1用のNewBASICが発表されたのもちょうどこの頃です。いま考えると，8ビットコンピュータがいちばん活気づいていた時期ではなかったでしょうか。

当時の私はというと，まだOh!MZの読者ではなくこのTHE SENTINELのページの存在すら知らないというただのPCユーザーでした。そんな私がこのページの存在を知ったのはそれからちょうど1年後の1986年6月号です。影山・金子両氏によるPC-8801用のS-OS"SWORD"がソースリスト付きで掲載されていたのです。しかも，その前の号にはスクリーンエディタまで発表されていました。

この，ソースリスト付きでプログラムを掲載し，主役は読者で，皆でこのシステムを育て上げようというロールプレイングシステムの企画は私の心を強く打ちました。さっそく売れ残っていた（失礼！）前月号と共にOh!MZを買って家に帰り，私のPC-8001mkIIに移植をしたのでした。

さて，移植が完成したものの手元にはスクリーンエディタE-MATEとディスクエディタぐらいしかありません。これでは投稿したくても動作確認のしようがありません。そこで，私は東京ロードマップを片手に編集部へ持ち込み投稿をすることにしたのでした。

まだ市ヶ谷にあったOh!MZ編集部で，Oh!PC編集部から借りていただいたPC-8001mkIISRを使ってチェックを開始したのですが，どうもBASICインタプリタ（皆さんお馴染みのFuzzyBASICです）の動きがよくない。その日は，FuzzyBASICのプログラムをソースごといただいて家に帰りチェックし，翌日再び挑戦，みごと動いたのでした。

それから，いろいろと手を加えPCの内部ルーチンを一切使用しないで高速に動作するところまでもっていったのですが，結局その"SWORD"はボツになり，ほかの方の作ったオールRAM版PC-8801用"SWORD"を私が多少手を加えて，PC-8801への変更点と共にPC-8001用としてOh!MZ上に発表させていただきました（Oh!MZ1987年9月号）。

そのときいただいたFuzzyBASICのソースリストが，のちにFuzzyBASICコンパイラ（1987年6月号発表）を作る際，大いに役立とうとは，本人すら予想のできないことでした。

当時，私は中学3年生で学校帰りに持ち込みをやるには珍しかったのでしょうか。いまでも金子さんに会うたびに「青い学生服を着ていた頃の君を知っている」といわれます。青い学生服を着ていた私もいま19歳です。そんな当時からもう4年目のS-OSです。"MACE"の頃から数えれば，今月号でまるまる5年のS-OSなのです。

リロケータブルってなんでしょう?

さて，そんなS-OSの世界にも次のステップが，やってきました。それは，リロケータブルアセンブラの登場です。

アセンブラには，大きく分けて2つの種類があります。ひとつはアブソリュート（絶対値）アセンブラといって，ソースプログラム→オブジェクト（目的）プログラムの変換を直接に行うもので，ZEDAとかREDAなんかがこれにあたります。

これらのアセンブラは，小規模なプログラム開発にはたいへん適しており，MACRO-80（CP/M上の標準的なリロケータブルアセンブラ。いまでこそ2万円で買えるが，昔は6万円もした）よりも，ZEDA（Oh!MZ1985年7月号に発表されたアブソリュートアセンブラ，Oh!MZを買っていてなおかつ体力の消費をいとわなければタダ）のほうが使いやすいということさえありますが，大規模なプログラム開発には，やや不便です（なぜかは後述します）。

では，そのアブソリュートアセンブラではない他方はというとリロケータブルアセンブラといって，ソースファイル→リロケータブルファイル→オブジェクトファイルというあいだに1ステップ置いた流れの前半の部分を受け持ちます（ちなみに，後半を受け持つのはリロケータブルファイルをつなぐ（リンクする）という意味でリンカといいます）。

1ステップ置くことによって，手間は若干増えますが，大規模なプログラム開発を行うときには，このひと工夫が生きてくるのです。

さて，リロケータブル（Relocatable：再び置くことのできる→再配置可能という意味）ファイルというのは聞き慣れない言葉だと思うので，ここで説明しておきます。

いままで，S-OS上では大規模なプログラム開発を行うときはZEDAやREDAの分割アセンブルとか断片アセンブルモードを用いて行っていました。この方法だと1カ所定数を変更してもソースプログラム全部をアセンブルし直さなければなりませんでした。これは，大規模なプログラム開発を行ううえで大変な時間的なロスです。そこで，多少姑息なテクニックとして，

```
;文字列を読み飛ばすサブルーチン
skipstr:    LD      A,(HL)
            INC     HL
```

```
        AND     A
        RET     Z
        JP      skipstr
```
というようなサブルーチンは,

```
    skipstr:    DB      7EH,23H,0A7
H,0C8H,0C3H
                DW      skipstr
```
としておけば少しはアセンブルに要する時間が短縮され，ソースプログラムの占有するメモリも少なくてすみます。ここで少し考えると，1バイトを表すのに，

```
    DB  7EH
```
などとやっているのは，まだまだ非経済的です。そこで，ある取り決めによってさらに圧縮率を高めたのがリロケータブルファイルです。

さらに，上のプログラムをよく見ていただくとラベルについての処理(DW skipstrのところ）が（意図的に）まだなされていません。ここがミソです。このことによってこのサブルーチンはどのようなアドレスに置かれても（配置されても）正しく動作するわけです。ここらへんが，リロケータブル（くどいようですが，Relocatable：再び置くことのできる→再配置可能）ファイルと呼ばれているゆえんです。

蛇足ですが，

```
            ORG     3000H
    ;文字列を読み飛ばすサブルーチン
    skipstr:    LD      A,(HL)
                INC     HL
                AND     A
                RET     Z
                JP      skipstr
```
を，

```
    skipstr:    DB      7EH,23H,0A7
H,0C8H,0C3H,00H,30H
```
と変換するアセンブラをアブソリュート(絶対値）アセンブラと呼ぶのも同様に理解できるのではないかと思います。絶対ジャンプと相対ジャンプを知っていればわかると思いますが，絶対値というのは，

$$|-1| = 1$$

とかいうのではなくて，この場合は，配置されるアドレスによらないある確定した値のことです。

メリット盛り沢山：セグメント編

以上で，だいたいリロケータブルファイルのありがたみが，理解していただけたかと思いますが，ほかにもいくつかリロケータブルファイルを用いると有利な点があります（リロケータブルアセンブラの本質的な機能ではなく，付加的な機能ですが)。

まず，セグメントという概念が導入されたこと。セグメントというのは，この場合，i8086についているセグメントとはちょっと違います。プログラムにはアルゴリズムを表している部分とデータを表している部分があります（特にマシン語のプログラムでは露骨にそうなる）が，ときとして両者を分けたい場合があります。

1本のソースファイルからオブジェクトファイルを作る場合でしたら（アブソリュートアセンブラを用いてもリロケータブルアセンブラを用いても）ソースプログラム上で，そうなるように（つまりソースファイル上でワークエリアを後ろのほうに持っていくとかして）コーディングすればよいのですが，複数のリロケータブルファイルからオブジェクトファイルを作る場合はそのような手法は用いることができません。

そこでリンカに複数のPC（Programing Counter。一説によるとPosition Counter）を持たせて，アルゴリズムを表しているコードがきたらコードセグメントを受け持つPCを用いて，データの部分はデータセグメントを受け持つPCを用いて処理してやります。具体的な例を挙げると，

```
    ;メインルーチン
            CSEG
    ;コードセグメントにセット
      MAIN:  LD      (SPBUF),SP
             CALL    なんらかの処理
             LD      SP,(SPBUF)
             RET

             DSEG
    ;データセグメントへ切り替え
      SPBUF:DS      2
```
というプログラムと，

```
    ;与えられた数の最大値を記憶する
            CSEG
      MAX:   LD      DE,(WKMAX)
             PUSH    HL
             OR      A
             SBC     HL,DE
             POP     HL
             RET     C
             LD      (WKMAX),HL
             RET

             DSEG
      WKMAX:DS      2
```
というプログラムをただ繋ぎ合わせると，ワークのSPBUFはプログラムとプログラムのあいだに埋もれてしまいます。もしこれが，ROMに焼き込まれるプログラムだとしたらものすごくまずいわけです（理由はわかるでしょう。ROM→Read Only Memory→書き込みできない→……)。

このとき，コードセグメントはアドレスの0番地から，データセグメントは100H番地からと，リンカに指示してやると，

```
    0000 ED 73 00 01  MAIN:
                        LD (SPBUF),SP
    0004 CD xx xx       CALL 処理
    0007 ED 7B 00 01    LD SP,(SPBUF)
    000B C9             RET
    000C ED 5B 02 01  MAX:
                        LD DE,(WKMAX)
    0010 E5             PUSH HL
    0011 B7             OR A
    0012 ED 52          SBC HL,DE
    0014 E1             POP HL
    0015 D8             RET C
    0016 22 02 01       LD(WKMAX),HL
    0019 C9             RET

    0100 00 00        SPBUF:DS 2
    0102 00 00        WKMAX:DS 2
```
となって無事に動作するようになるわけです。

掲載されるプログラムでもこのようにワークエリアとプログラム（コードセグメント内にある）を分離・分割してやるとワークエリアの分だけ読者に打ち込んでもらう量が少なくてすみます。

メリット・メリット：ライブラリ編

プログラミングをしていると，同じサブルーチンをいろいろなプログラムで使うことがあります(VRAMのアドレス計算ルーチンだとか，10進数表示ルーチンだとか)。そのようなルーチンはひとつのファイルにまとめておきコンピュータに適当に取捨選択させておけば便利です。これがライブラリの考え方であり，その役割を果たすのがライブラリアンです。

たとえば「私，元気に生きてる」という手紙(ここではプログラムのアナロジーね）を出す場合，たとえ相手が親であってもこれだけではあまりにもぶっきらぼうで失礼です（と，広く世間ではいわれている)。

そこで，この手紙をライブラリアンにかけると自動的に「拝啓，目にしみるような青葉に風わたるところとなり……」とか，「暑さ寒さも彼岸までとか……」というような，一度作ってしまえば二度と変えない＆よく使うような文（サブルーチン）を適宜

表1 リロケータブルフォーマットの仕様 表中において，n：2バイトの数値（下位バイト，上位バイトの順）

［00H＋ラベルの文字数－1（0～31）］＋［ラベルナンバー（2バイト）］＋“ラベル名”　外部ラベルの定義
［20H＋ラベルの文字数－1（0～31）］＋“ラベル名”　外部ラベルの参照
［60H＋（データの数－1（0～31））］＋［データ1］＋［データ2］＋…………　オブジェクトコードの出力
※）“ラベル名”とはラベルの名前のASCII形式文字列そのものです。エンドコードなどは一切含みませんので注意してください。

80H, n
　ラベルナンバーnの値をスタックへプッシュ
81H, n
　スタック上の値（2バイト）を取り，ラベルナンバーnのラベルに定義する
82H
　スタック上の値（2バイト）を下位8，上位8バイトの順にオブジェクトファイルに出力
83H
　スタック上の値（2バイト）を上位8，下位8バイトの順にオブジェクトファイルに出力
84H
　スタック上の値（2バイト）を取り，下位8バイトのみをオブジェクトファイルに出力
85H
　スタック上の値（2バイト）を取り，現在のPCとの差をとり下位8バイトのみをオブジェクトファイルに出力（JRやDJNZのオペランド用）
86H, n
　現在のPCにn（2バイト）を加えて，その値をオブジェクトファイルに出力
87H
　現在のPCをスタックに積む
90H＋m，データ1，データ2，………
　データ（1バイト）をm＋1個，そのままオブジェクトファイルに出力（DEFBに対応）
98H＋m，データ1，データ2，………
　データ（2バイト）をm＋1個，下位，上位の順にオブジェクトファイルに出力（DEFWに対応）
A0H＋m，データ1，データ2，………
　データ（2バイト）をm＋1個，上位，下位の順にオブジェクトファイルに出力（FDBに対応）
A8H＋m“8文字までの文字列”
　（m＋1）文字からなる文字列をそのままオブジェクトファイルに出力する（DEFM“～”に対応）
　（なお，9文字以上の文字列は，8文字ごとに分割する ）
アイテム90HからA8Hまでにおいて　0≦m≦7
B0H
　スタック上の値を取り，その値の分だけPCを進める（DEFSに対応）
C0H ＋　C4H HIGH　C8H^　CCH　AND
C1H －（2項）C5H LOW　C9H右シフト　CDH OR
C2H ＊　C6H －（単項）　CAH左シフト　CEH XOR
C3H ／　C7H MOD　CBH NOT　CFH 上位8バイトと下位8バイトの入れ替え
　※これらのアイテムは，リンカの演算スタックに対して，FORTH風に操作を行う
E0H, n，“文字列”，00H
　ライブラリファイルのヘッダ部分に用いられるアイテム
　シークアドレスn以降から，アイテムFFHが存在するアドレスまでモジュール名“文字列”のモジュールであることを示す
E1H, m（1バイト）
　セグメント指定　m＝0 コード用　m＝1 データ用　m＝2 ワーク用
E2H
　スタック上の値を取り，その値をオペランドとしてPHASE文を実行
E3H
　DEPHASE文
E4H
　スタック上の値を取り，下位8バイトのみ出力　（DEFB　数式　に対応）
E5H
　スタック上の値を取り，下位，上位の順に出力　（DEFB　数式　に対応）
E6H
　スタック上の値を取り，上位，下位の順に出力　（FDB　数式　に対応）
E7H, n
　スタック上にnを積む
E8H
　スタック上の値を取ってORG命令のオペランドとする
E9H
　スタック上の値を取ってEXADR（実行開始アドレス）に指定する
FBH，“文字列”＋00H
　リンク時，ディスプレイ上に“文字列”を表示する
FCH
　スタック上の値をディスプレイ上に10進数で表示
FDH
　スタック上の値をディスプレイ上に16進4桁で表示
FEH
　スタック上の値をディスプレイ上に16進2桁で表示
FFH
　ファイル，モジュール，ライブラリ・ファイルのヘッダ部の終わり

付け足していってくれるわけです。

で，私もよくは知らないのですが，相手とか状況によって「拝啓」が「謹啓」だったり「拝呈」だったり「一筆申し上げます」だったりします。そこらへんもライブラリアンが面倒をみてくれます。つまりプログラマ（手紙を書く人）は，プログラム（手紙）の書きたいところ（手紙の本文ですな）だけに専念していればあとの付属的なサブルーチン（時候の挨拶など）は一度，作ってしまいさえすればなにも考えなくてすむのです。

C言語を使ったことのある人なら，printfはC言語の持っているステートメントではなく，誰かが作ってあらかじめライブラリファイルに収めておいてくれたサブルーチンだってのは知ってますよね。

さぁーて，来月のS-OSは

来月号のこのコーナーには，いま述べたようなリロケータブルアセンブラを発表する予定です。ネーミングについてはいろいろ考えたのですが，結局こんなところに落ち着きました。WZD（ダブルゼーダ）です。まぁ，いろいろ憶測もあるかと思いますが，とりあえずアセンブラとリンカ（ちなみにWLKという）2つで（ダブルで）ZEDA（ゼーダ）の役割，と思ってください。

さて，このWZDでは表1に示したリロケータブルフォーマットを使用しています。このファイル形式は以下のような特徴を持っています。

1)　疑似命令で出力されるオブジェクトに対するアイテムと，ニーモニックから出力されるオブジェクトに対するアイテムが別である。
2)　ラベルナンバーという新しい機能を持っている。
3)　まだまだ，未定義なアイテムが多い（拡張性はロールプレイングシステムの必須条件！）。
4)　リンキングローダに対してバックパッチ（チェーンパッチ）を命令するアイテムがない。
5)　ビット単位でなくバイト単位でアイテムを定義しているので，処理は速いが圧縮率は低い（正確には知らないが多分MACRO-80の5割増しくらい）。
6)　ライブラリファイルはそのモジュール内に同じ名前の外部ラベルがなくてはならない。
7)　S-OS上のシステムなのに68系のオブジェクトにも対応できるように考慮してあ

る（あくまでも考慮だけど……）。
8） リンク作業中，コンソールに表示ができる。

さて，これだけではわかりづらいでしょうから順に解説しましょう。1)はどういうことかというと，

```
    LD  A, 30H
```

と，

```
    DB  3EH, 30H
```

は，オブジェクトにしたら同じですがリロケータブルファイルのあいだは違う意味を持たせてやろうということです。このことによってリロケータブルファイルからなら，どこがデータで，どこがコードか，という情報を逆アセンブラに知らせてやらなくても自動的に逆アセンブルを行えます（そういう逆アセンブラを計画しています。どうぞご期待ください）。

ただし，この機能は圧縮率を優先してWZDには持たせてありません。まあ，WZDからリロケータブルファイルが出力されるということは，ソースファイルが手元に存在するということですので逆アセンブルの必要性はない，ということで勘弁してください。

2)に出てくるラベルナンバーというのは皆さんは聞いたことがないと思いますが，聞いたことがあったら変です。なんせ私の造語ですから。どういうものかというとFORTRANやCコンパイラの吐き出すアセンブルファイルにはよく，

```
   @19：LD   A, H
        OR   L
        JP   Z, @12
        JP   @14
```

というふうに，@19とか@12のようなラベルが出てきます。これはwhile文やDO文の飛び先に使われるのです。

これは番地振り付けをアセンブラにしてもらって，コンパイラ自身の負担を減らそうとしているのです。そこで，このようなラベルをリンカのほうでも面倒を見てやれるようにすれば，コンパイラは直接リロケータブルファイルを吐き出せるのではないか，という発想からこのような機能をサポートするアイテムを用意して，これをラベルナンバーと名づけました（いま考えると，ナンバードラベルのほうがよかった気もする）。

そして，さらにさらに便利なことに1)の機能と組み合わせることによって必要とあらば，コンパイラの出力したリロケータブルファイルから，コンパイラが出力するであろうアセンブルファイルがほぼ完全なかたちで再現できてしまうのです。

4)はリンキングローダの動作を知らないと説明しづらいのですが，このことによって１パスの高速リンカを作るときにたいへん不都合だといっておきます。これは，そんなことをするなといっているのではなく１パスの高速リンキングローダを作ってくれる人に（これを読んでいるそこのあなた，あなたですよっ！ っていうのは，もう古いかな？）適当に未定義アイテムを割り振ってくださいということなのです。

いまのところ私には，リンキングローダを作る予定はないし，どういうものが適当かというヴィジョンもないので，リンケージエディタを作ってくれるあなたはここらへんで，おおいに創意工夫をこらしてください。

6)は，たとえばライブラリ中のprintfというモジュール内には必ずprintfという外部ラベルがなきゃだめ，ということです（ふつうは，ライブラリファイルの中全体をリンカがくまなく探してくれるんだけどあえてそうはしないで，ライブラリファイルの先頭にインデックスをつけるという構造をとった）。

正確にいうと，なくてもかまわないのですが，かなり高等なテクニックなのでまたの機会に説明します（Small-Cのライブラリにはこのテクニックが10カ所ぐらい使われているので，そのときにでも）。

7)はただ，16ビットの数値を上位，下位の順にも出力できるというだけのことです(80系の場合は下位，上位の順)。

8)は特別なファイルをリンクするときに
「このファイルは，スタートアドレスをｘｘ00Hにしないと動作しない」
とか，
「このファイルの後ろには，すまないがｘｘｘｘHバイトほど空けておいてくれ」
とか，表示できるようにするための機能ですが，ほかのシステムではあまり見かけないのでどのように使ったらよいのかアドバイスできません。なにか面白い使い方があるのではなかろうかと思い用意してみました。

また，1987年11月号で発表されたファイルアロケータ＆ローダのフォーマットとは，似ても似つかぬフォーマットですが，それには，ちゃんと抜け道を用意しています。リンカの発表のときにでも，教えますので，心配しないで，ご期待ください。

急募

逆アセンブラ作成のため，逆ポーランド式から普通の算出式に変換するアルゴリズムを募集します。履歴書・経験不要，男女，学歴その他一切不問，連絡は担当石上まで，てなわけで，皆さんよろしくお願いします(逆アセンブラそのものを作ってくれるって名乗りをあげてくれるともっと嬉しいけど)。どの本に載っているとかいう情報でも嬉っしいなっ！ 理論的にそんな変換は不可能だという情報であっても悲しいけどありがたい。

最後に

なぜ，遊びも仕事も16ビット機の時代に８ビット機でこんなことをして喜んでいるのかと，不思議に思う方もいるのではないかと思います。自分でも16ビット機を持っていて，なぜこんなにも８ビットの世界に閉じこもっているのか，答えが見つかりません。大学に受かっても予備校のテキストやノートを捨てられない心境に似ているかもしれません（仮面浪人というのは，この際考えない）。

それとも，私がただ16ビットの世界を知らないだけかもしれません。

ただ，X68000やPC-9801では，いまのところ私の居場所が見あたらないので，もうしばらくS-OSシステムの作成を続けようと思います。

８ビット機の記事の締めくくりは，いつもワンパターンでいやなのですが，やはりいっておきます。

まだまだ８ビットは捨てたものじゃないっ！ では，来月WZDの記事でお会いしましょう。

さてさて，突然ですが

私はコール・ポリフォニーという合唱サークルに入っているのですが，そこのコンサートを大東文化大学混声合唱団，東洋大学混声合唱団とジョイントで下記のとおり行います。曲は，「祈祷天昇」というもので，関東地方ではまだ演奏されていないものだそうです。

音楽・合唱に興味のある方，板橋文化会館に興味のある方，大学のサークルとはどんなところか知りたい方，その他の方々全員の，ご来場を心からお待ちしております（このコンサートが（企画面で）失敗するようなことがあれば，私は，来月からプログラムができなくなってしまふ。１カ月ぐらい，東京湾の湾岸警備かなぁ……）。

第10回　林鐘会演奏会
日時/６月16日（土）
開場/17：30　開演/18：00
場所/ 板橋文化会館
料金/600円

全機種共通S-OS"SWORD"要

STACK用ゲームSQUASH!

先月発表したインタプリタ言語STACK用のサンプルプログラム。3Dテニスゲームsquash!です。インタプリタでもそこそこの速度ですがセミコンパイルしてお楽しみください。実行には各機種用3DグラフィックパッケージMAGICが必要です。

Yamada Junji
山田 純二

5月号に掲載されたSTACK用のゲームです。すでに常連と化してしまった平井氏の作品で，平井氏とは切っても切れない縁となっている私が，予想どおりサンプルのゲームを作ることになりました。ゲーム内容は，せっかくMAGICをサポートしているんだからなんか3Dでゲームを作ろう，と思ってできたものが，この3D版テニスといういかにもサンプルらしいものです。

このSTACKという言語はFORTHとBASICの性格をあわせた，とてもユニークな言語で，日頃スタック型言語に馴染みのないユーザーの人たちは，仕様書を見ただけでは理解しづらいでしょう。

そんな人はこのリストを見ながら使い方を学んでいってください。それと，このゲームはMAGICを使っていますので，MAGICを持っていない人，もしくは対応していないユーザーの人はごめんなさい。G-RAMは3ページ分使っていますが，1ページでもなんとかできるかもしれませんので，力が余っている人はチャレンジしてください。

入力方法

まずは，オブジェクトデータをMACHINTO-Cなどのツールを使って入力します。チェックサムを確認して間違いがないかどうか確かめたあと，

START ADRESS : 9000H
END ADRESS : 90D8H

エントリーアドレスは省略して，まずはデバイスにセーブしましょう。

次にメインのリスト1をSTACKのエディタ上から入力してください。とりあえずはオブジェクトデータを読み込んだあとGコマンドでインタプリタ上から実行できます。

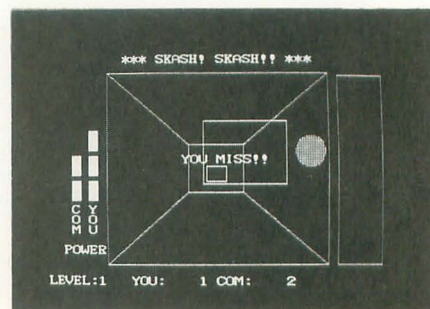
しかし，このままだと実行速度がいまいちなので，エラーがないことが確認されたら，セミコンパイラ機能を使ってコンパイルすることをすすめます。コンパイルアドレスは8000Hにすると，ちょうどデータの少し手前でコンパイルが終わりますので，オブジェクトをデータとまとめてセーブしておけば次からは，いちいちコンパイルしなくてもSTACK上からJ8000で実行されます。もちろん実行する前に，ちゃんとMAGICをロードしておくのを忘れずに。

遊び方

ルールは普通のテニスゲームと同じくコンピュータから10ポイント先取すればプレイヤーの勝ち！　で，次のレベルのコンピュータと対戦していきレベル5のコンピュータに勝つか，コンピュータに10ポイント先取されると，ゲームオーバーになるというものです。

主画面は左からパワーメーター，ゲーム画面，いちばん右がゲーム画面を上から見た図，となっています。左下にはコンピュータのレベル，ゲーム画面の下には得点が表示されます。左側にあるパワーメーターというのは，ボールがパットに当たったときの跳ね返り速度を決めるものです。メーターが2のときには速度を変化させず，1のときには減速，3のときにはボールの速度を加速させます。

コンピュータのレベルは1～5まであリますが，はっきりいってあまり強くありませんので，不満があれば自分でなんとかしましょう。

操作方法はKを中心としたキーでパットの上下左右の移動，スペースキーでパワーメーターを操作します。メーターは1回押すたびにひとつずつ上がって，3つ目まで

いくとひとつ目に戻ります。あと，自分は4方向にしか動けないのに，相手は8方向に動けるなんてひきょうじゃないか，と思われるでしょうが，これはプレイヤー側のハンデと思って見逃してくださいね。

ボールの表示はまず，中心座標と半径の分を足したダミーデータをMAGICで3Dから2D変換を行ってから，座標を取り出して円の塗り潰しを行うことによって実現しています。

プログラムは，はっきりいって見にくいです。特に，数式が逆ポーランド記法にいったんなおしてから記述しなければならないので，知らない人にはなにがなんだかわからないかもしれません。慣れるまでリストを見たり，仕様書を見たりリストを読む

表1　主な変数表

変数	内容
D1, D2, D3	ボールの移動量
P1	プレイヤーのパワー
P2	コンピュータのパワー
S1	プレイヤーのスコア
S2	コンピュータのスコア
L1	コンピュータのレベル
R1	コンピュータの進行方向
X1, Y1	プレイヤーのX，Y座標
X2, Y2	コンピュータのX，Y座標

MAGICパラメータアドレス

アドレス	内容
M1	プレイヤーのパット
M2	コンピュータのパット
M3	フィールドのBOX
M4	消去データ
M5	ボールダミーデータ

のもひと苦労かもしれません。がんばってください。

リストにはできるだけ注釈を付けておきましたし，主な変数表も付けておきますのでリストの改造をしたいときなんかには，参考にしてください。

ツールを使いこなそう

S-OSの企画が始まりいままでにずいぶんいろいろなツールが発表されてきましたが，それらを使ったアプリケーションがほとんどないのはどういうことでしょう。

それぞれ，作者の趣味でくせがあったりして，とっつきにくい面があるかもしれませんが，あまり大規模なプログラムを作らないときにはアセンブラなどで作るより，はるかに開発効率がいいはず。S-OSはユーザー自身の手で育てていく，という精神を皆，忘れてしまったのでしょうか。口だけで「もっと便利なツールがほしい！」などといっている暇があったら，いままでに発表されたものを使ってなんとか自分好みのものを作ってやろう，というぐらいの気迫を持ってがんばりましょう。読者が参加しないとどうしようもないんですから。

私もえらそうなことをいえるほど実力があるわけではありませんが，他人をあてにせず自分自身も精進してバリバリプログラムを組みましょう。

リスト1

```
9000 0B 00 00 00 0B 01 00 00 : 17
9008 0B 02 00 00 0B 03 00 00 : 1B
9010 0B 04 00 00 0B 05 00 00 : 1F
9018 0B 06 00 00 0B 07 00 00 : 23
9020 0B 08 00 00 0C 08 D8 FF : FE
9028 E2 FF 00 00 28 00 E2 FF : EA
9030 00 00 D8 FF 1E 00 00 00 : F5
9038 28 00 1E 00 00 00 D8 FF : 1D
9040 E2 FF 96 00 28 00 E2 FF : 80
9048 96 00 D8 FF 1E 00 96 00 : 21
9050 28 00 1E 00 96 00 0C 00 : E8
9058 01 01 03 03 02 02 00 04 : 10
9060 05 05 07 07 06 06 04 00 : 28
9068 04 01 05 02 06 03 07 0D : 29
9070 0E 0F 0B 00 00 00 0B 01 : 34
9078 00 00 0B 02 00 00 0F 0B : 27
---------------------------------
SUM: F9 28 A7 0C 68 23 3B 19 428F
```

```
9080 00 F6 FF 0B 01 00 00 0B : 0C
9088 02 96 00 0F 0C 04 F1 FF : A7
9090 F6 FF 00 00 F1 FF 0A 00 : EF
9098 00 00 0F 00 0A 00 00 00 : 19
90A0 0F 00 F6 FF 00 00 04 00 : 08
90A8 01 01 02 02 03 03 00 0D : 19
90B0 0E 0F 07 00 00 0E 07 02 : 3B
90B8 00 0F 0B 00 00 00 0B 01 : 26
90C0 00 00 0B 02 00 00 0C 02 : 1B
90C8 00 00 00 00 00 00 05 00 : 05
90D0 00 00 00 00 01 00 01 0D : 0F
90D8 0F 00 00 00 00 00 00 00 : 0F
90E0 00 00 00 00 00 00 00 00 : 00
90E8 00 00 00 00 00 00 00 00 : 00
90F0 00 00 00 00 00 00 00 00 : 00
90F8 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 25 AA 23 1D 0C 14 23 29 11F7
```

リスト2

```
  1
  2 ;
  3 ;SQUASH! SQUASH!!
  4 ;       FOR STACK
  5 ;   1990 4.1  by J.Yamada
  6
  7    40 WIDCH
  8     INIT  0 2 COL
  9      2 1  .L1 .R1         ;COM LEVEL
 10      0 0  .S1 .S2
 11      "C" COTR
 12       GOSUB 199
 13       GOSUB 30
 14 %5
 15      "C" COTR
 16      1 .F1  0 .S1 0 .S2
 17         GOSUB 199
 18 %10
 19      BREAK
 20      GOSUB 1              ;MY PAD
 21      GOSUB 300            ;COMPUTER PAD
 22      GOSUB 50             ;BOLL MOVE
 23      #H1 =0 IF GOTO 10
 24      GOSUB 55
 25      #S1 9  > #S2 9  > OR IF GOTO 20
 26      GOTO 10
 27 %20
 28     #S2 9  > IF GOTO 21
 29     2 BELL
 30      16 13 LOCATE "YOU WIN!" PRTS
 31      5 #L1 == IF GOTO 22
 32      11 15 LOCATE "NEXT COMPUTER LEVEL:"
 33      INC L1
 34      PRTS #L1 PRINT1
 35     GOSUB 30
 36     GOTO 5
 37 %22
 38     12 15 LOCATE "*** GAME OVER ***"
 39     PRTS
 40    15 17 LOCATE "RETRY(Y/N)?"
 41    PRTS
 42    FLGET .K #K "Y" ASCII == IF #K CHR 1 .L1 GOTO 5
 43    #K "N" ASCII != IF GOTO 22
 44    #K CHR
 45   END
 46
 47 ;YOU LOST
 48 %21
 49      16 13 LOCATE "YOU LOST!" PRTS
 50      12 15 LOCATE "TRY AGAIN(Y/N)?" PRTS
 51      FLGET .K #K "Y" ASCII == IF #K CHR GOTO 5
 52      #K "N" ASCII != IF GOTO 21
 53      #K CHR
 54     END
 55
 56 ;PUSH ANY KEY!
 57 %30
 58     13 17 LOCATE
 59       "PUSH ANY KEY!!" PRTS
 60     13 17 LOCATE
 61       "                " PRTS
 62       GETKEY =0 IF GOTO 30
 63 %31
 64       GETKEY 0 != IF GOTO 31
 65   RET
 66
 67 ;MY PAD MAIN
 68 %1
 69      GETKEY .B  ;#B =0 IF GOSUB 191 RET
 70       #M1 2 + PEEKW .X1
 71       #M1 6 + PEEKW .Y1
 72     #B "J" ASCII == IF GOSUB 11    ;LEFT
 73     #B "L" ASCII == IF GOSUB 12    ;RIGHT
 74     #B "M" ASCII == IF GOSUB 13    ;DOWN
 75     #B "I" ASCII == IF GOSUB 14    ;UP
 76     #B " " ASCII == IF GOSUB 45    ;POWER
 77 %2
 78      GOSUB 195
 79      #X1 #M1 2 + POKEW
 80      #Y1 #M1 6 + POKEW
 81      GOSUB 191
 82    RET
 83
 84 %11
 85      #X1 -20 F< IF  RET
 86       #X1 5 - .X1
 87   RET
 88
 89 %12
 90      20 #X1 F< IF  RET
 91       #X1 5 + .X1
 92   RET
 93
 94 %13
 95     15 #Y1 F< IF  RET
 96       #Y1 5 + .Y1
 97   RET
 98
 99 %14
100      #Y1 -15 F< IF  RET
101       #Y1 5 - .Y1
102   RET
103
104 %45
105      INC P1 #P1 4 == IF 1 .P1
106    GOSUB 251
107   RET
108
109 ;BOLL MAIN
110 %50
111   0  .H1       ;HANTEI FLAG
112    GOSUB 202
113    #M5 2 + PEEKW .X3
114    #M5 6 + PEEKW .Y3
115    #M5 10 + PEEKW .Z3
116    #Z3 #D3 +  0 F< IF  #D3 NEGATE .D3 GOSUB 100
117     #H1 $FF == IF RET
118    150 #Z3 #D3 +  F< IF GOSUB 101 #D3 NEGATE .D3
119     #H1 $50 == IF RET
120    40 #X3 #D1 +  F< #X3 #D1 + -40 F< OR IF #D1 NEGATE .D1
```

```
121     25 #Y3 #D2 +  F< #Y3 #D2 + -25 F< OR IF #D2 NEGATE .D2
122        #X3 #D1 + .X3
123        #Y3 #D2 + .Y3
124        #Z3 #D3 + .Z3
125     GOSUB 196
126   #X3 #M5  2 + POKEW
127   #Y3 #M5  6 + POKEW
128   #Z3 #M5 10 + POKEW
129   GOSUB 193
130   GOSUB 201
131  RET
132
133 ;MISS!!
134 %55
135   #H1 $FF == IF "YOU MISS!!" INC S2 GOTO 56
136    "COM MISS!!" INC S1
137 %56
138    16 11 LOCATE PRTS
139    GOSUB 260 1 BELL
140    #F1 NEGATE .F1
141
142    GOSUB 195 GOSUB 196
143    GOSUB 197 GOSUB 202
144
145    GOSUB 199
146    16 11 LOCATE "            " PRTS
147    0 .H1
148   RET
149
150
151 ;ATARI? MY PAD
152 %100
153     #M1 2 + PEEKW .W3
154     #M1 6 + PEEKW .W4
155     #P1 .W5
156        GOSUB 105
157    RET
158
159 ;ATARI? COM PAD
160 %101
161     #M2 2 + PEEKW .W3
162     #M2 6 + PEEKW .W4
163     #P2 .W5
164        GOSUB 105
165        #H1 0 != IF $50 .H1
166    RET
167
168 %105
169     #W3 #M5 2 + PEEKW - .W1
170        #W1 -19 F< 19 #W1 F< OR IF $FF .H1
171     #W4 #M5 6 + PEEKW - .W2
172        #W2 -14 F< 14 #W2 F< OR IF $FF .H1
173        #H1 0 != IF RET
174
175 ;HANEKAERI
176     8  #W1 F< IF #D1 3 - .D1
177     #W1 -8 F< IF #D1 3 + .D1
178     3  #W2 F< IF #D2 3 - .D2
179     #W1 -3 F< IF #D2 3 + .D2
180
181      9 #D1 F< IF  9 .D1
182     #D1 -9 F< IF -9 .D1
183      9 #D2 F< IF  9 .D2
184     #D2 -9 F< IF -9 .D2
185
186   #W5 2 == IF RET
187   #W5 3 == IF #D3 3 + .D3
188   #W5 1 == IF #D3 3 - .D3
189    #D3 15 == IF 9 .D3
190    #D3  0 == IF 3 .D3
191  RET
192
193
194 ;BOX PRINT
195
196 %190
197    1 2 COL
198     #M3 MAGIC
199    0 2 COL
200   RET
201
202 ;MY PAD PRINT
203 %191
204       #M1 MAGIC
205        #O1 MAGIC
206          $C2B4 PEEKW .X9
207          $C2B6 PEEKW .Y9
208          $C2BC PEEKW .X
209          $C2BE PEEKW .Y
210      RET
211
212 ;COMPUTER PAD PRINT
213 %192
214     #M2 MAGIC
215      #O1 MAGIC
216          $C2B4 PEEKW .X7
217          $C2B6 PEEKW .Y7
218          $C2BC PEEKW .X8
219          $C2BE PEEKW .Y8
220  RET
221
```

```
222 ;BALL PRINT
223 %193
224     $AA33 $AA33 TILE
225 %194
226     2 2 COL
227     #M5 MAGIC
228       $C2B4 PEEKW .X5
229       $C2B6 PEEKW .Y5
230       $C2B8 PEEKW .X6
231       $C2BA PEEKW .Y6
232     #X5 #Y5 #X6 #X5 - CIRCLE
233     0 2 COL
234    RET
235
236 ;BALL ERASE
237 %196
238    0 0 TILE  GOTO 194
239
240 ;BOX ERASE
241 %195
242     0 0 COL
243      #X9 #Y9 #X  #Y  BOX@
244      0 2 COL
245     RET
246
247 ;BOX ERA 2
248 %197
249 ;  #M4  MAGIC
250     0 0 COL
251      #X8 #Y8 #X7 #Y7  BOX@
252      0 2 COL
253     RET
254
255 ;SCREEN INIT
256
257 %199
258    06 .D3
259    #F1 1 != IF #D3 NEGATE .D3
260     RND 3 MOD 1 + 3 * .D1
261     RND 3 MOD 1 + 3 * .D2
262      9 1 LOCATE
263     "*** SKASH! SKASH!! ***"
264      PRTS
265      2 20 LOCATE "POWER" PRTS
266      02 02   .P1 .P2
267      $908C .O1  $9072 .M1  ;MYPAD
268                 $907F .M2  ;COMPAD
269                 $9000 .M3  ;BOX
270                 $90B2 .M4  ;ERASE DATA
271                 $90BA .M5  ;CIRCLE DATA
272        0 0 0 #M1 2 + POKEW
273              #M1 6 + POKEW
274              #M1 10 + POKEW
275
276     150 0 0 #M2 2 + POKEW
277             #M2 6 + POKEW
278             #M2 10 + POKEW
279
280      75 0 0 #M5 2 + POKEW
281             #M5 6 + POKEW
282             #M5 10 + POKEW
283
284     GOSUB 200
285     GOSUB 251
286     GOSUB 252
287     GOSUB 190
288     GOSUB 191
289     GOSUB 192
290     GOSUB 193
291     GOSUB 260
292     GOSUB 270
293    RET
294
295 ;BALL PRINT
296 %200
297    1 2 COL
298     540 25 620 175 BOX@
299    0 2 COL
300 %201
301    $FFFF $FFFF TILE
302 %203
303    #M5 2 + PEEKW .W1
304    #M5 10 + PEEKW .W2
305     #W1 580 + 175 #W2 - 8 CIRCLE
306   RET
307 %202
308    0 0 TILE GOTO 203
309
310 ;POWER METER PRINT
311 %250
312      REPEAT
313        $FFFF $FFFF TILE
314        #W1 16 - #W2 15 - #W1 #W2 BOXFUL
315        #W2 20 - .W2
316        DEC W3 #W3 =0 UNTIL
317         3 #W4 - .W4 #W4 =0 IF RET
318      REPEAT
319       0 0 TILE
320       #W1 16 - #W2 15 - #W1 #W2 BOXFUL
321       #W2 20 - .W2
322       DEC W4 #W4 =0 UNTIL
```

▶俺のX1Cは家にきて7年目になった。本体の色が落ちてきて渋くなった。
佐竹　孝洋（18）神奈川県

```
323   RET
324
325 ;MY POWER
326 %251
327     96 124 .W2 .W1
328     #P1 #P1 .W3 .W4
329      GOSUB 250  5 16 LOCATE
330       "Y" PRTS "LD" COTR "O" PRTS
331       "LD" COTR "U" PRTS
332   RET
333
334 ;COM POWER
335 %252
336     64 124 .W2 .W1
337     #P2 #P2 .W3 .W4
338      GOSUB 250  3 16 LOCATE
339       "C" PRTS "LD" COTR "O" PRTS
340       "LD" COTR "M" PRTS
341   RET
342
343 ;SCORE PRINT
344 %260
345     10 23 LOCATE
346      "YOU:" PRTS #S1 PRINT
347      " COM:" PRTS #S2 PRINT
348   RET
349
350 ;COM LEVEL PRINT
351 %270
352       0 23 LOCATE "LEVEL:" PRTS
353     #L1 PRINT1
354  RET
355
356 ;COMPUTER MAIN
357 %300
358       #M2 2 + PEEKW .X2
359       #M2 6 + PEEKW .Y2
360        #L1 01 == IF GOSUB 350
361        #L1 02 == IF GOSUB 360
362        #L1 03 == IF GOSUB 370
363        #L1 04 == IF GOSUB 380
364        #L1 05 == IF GOSUB 390
365     #X2 #M2 2 + POKEW
366     #Y2 #M2 6 + POKEW
367     GOSUB 197
368     GOSUB 192
369     RET
370
371 ;COM XY CHECK
372 %310
373   0 .H2
374     #R1 1 == IF  GOSUB 314  ;UP
375     #R1 2 == IF  GOSUB 311  ;LEFT
376     #R1 3 == IF  GOSUB 313  ;DOWN
377     #R1 4 == IF  GOSUB 312  ;RIGHT
378
379     #R1 5 == IF  GOSUB 315  ;RU
380     #R1 7 == IF  GOSUB 316  ;RD
381     #R1 9 == IF  GOSUB 317  ;LU
382     #R1 11 == IF GOSUB 318  ;LD
383  RET
384
385 %311
386     #X2 -20 F< IF -20 .X2 GOTO 343
387       #X2 #T1 - .X2
388   RET
389 %312
390     20 #X2 F< IF 20 .X2 GOTO 343
391       #X2 #T1 + .X2
392   RET
393 %313
394    15 #Y2 F< IF 15 .Y2 GOTO 343
395      #Y2 #T2 + .Y2
396   RET
397 %314
398     #Y2 -15 F< IF -15 .Y2 GOTO 343
399       #Y2 #T2 - .Y2
400   RET
401 %315
402    GOSUB 312 GOSUB 314
403   RET
404 %316
405    GOSUB 312 GOSUB 313
406   RET
407 %317
408    GOSUB 311 GOSUB 314
409   RET
410 %318
411    GOSUB 311 GOSUB 313
412   RET
413
414 ;OVER
415 %343
416   $FF .H2  RET
417
418 ;COM LEVEL1
419 %350
420    3 3 .T1 .T2
421    #M5 10 + PEEKW 85 F< IF RET
422     GOSUB 400
423     GOSUB 310
424   RET
425
426 ;COM LEVEL2
427 %360
428    3 5 .T1 .T2
429    #M5 10 + PEEKW 75 F< IF GOTO 361
430     GOSUB 400
431     GOSUB 310
432   RET
433 %361
434     #R1 2 == #R1 4 == OR IF GOTO 362
435     2 .R1
436 %362
437     GOSUB 310
438     #H2  =0 IF RET
439     #R1 2 == IF 4 .R1 RET
440     2 .R1
441  RET
442
443 ;COM LEVEL3
444 %370
445    5 5 .T1 .T2
446    #M5 10 + PEEKW 65 F< IF GOTO 371
447     GOSUB 400
448     GOSUB 310
449     #D3 9 F< IF 2 .P2 GOTO 373
450     1 .P2
451 %373
452     GOSUB 252
453   RET
454 %371
455     #R1 1 == #R1 3 == OR IF GOTO 372
456     1 .R1
457 %372
458     GOSUB 310
459     #H2  =0 IF RET
460     #R1 1 == IF 3 .R1 RET
461     1 .R1
462  RET
463
464 ;COM LEVEL4
465 %380
466    7 7 .T1 .T2
467    #M5 10 + PEEKW 50 F< IF RET
468     GOSUB 400
469     GOSUB 310
470     3 .P2
471     GOSUB 252
472   RET
473
474 ;COM LEVEL5
475 %390
476    8 8 .T1 .T2
477     GOSUB 400
478     GOSUB 310
479    GOSUB 410
480  RET
481
482 ;BALL OIKAKE
483 %400
484    0 0 .W3 .W4
485      #M2 2 + PEEKW
486      #M5 2 + PEEKW - .W1
487        #W1 =0 IF GOTO 401
488        0 #W1 F< IF 4 .W3 GOTO 401
489        2 .W3
490 %401
491      #M2 6 + PEEKW
492      #M5 6 + PEEKW - .W2
493        #W2 =0 IF GOTO 402
494        0 #W2 F< IF 1 .W4 GOTO 402
495        3 .W4
496 %402
497      #W3 =0 #W4 =0 OR IF #W3 #W4 + .R1 RET
498      #W3 2 * #W4 + .R1
499    RET
500
501 ;POWER SET
502 %410
503     #D1 .W1 GOSUB 420
504      #W1 .W2
505     #D2 .W1 GOSUB 420
506      #W1 .W3
507     #D3 .W1 GOSUB 420
508      #W1 #W2 #W3 + +  .W3
509      2 .P2
510      #W3 18 F< IF INC P2
511      25 #W3 F< IF DEC P2
512   GOSUB 252
513  RET
514
515 ;INT(W1)
516 %420
517    0 #W1 F< IF RET
518     #W1 NEGATE .W1
519   RET
520
521   END
```


▶Oh！MZ　1987年10月号 BROAD SWORDで次の３バイトを書き換えると無敵になります。$9D7EからのCA, 9BをC3, 86に$A5B9のC2をC3にです。最初の面に戻るのに17分もかかる(つまりそれだけ遊べる)。
大橋　道雄 (33) 岐阜県

X68000用

S-OS“SWORD”

ついにX68000にもS-OS“SWORD”ができました。基本的にソフトウェアエミュレーションですが，内部ルーチンは68000ネイティブコードで記述されていますので高速。操作感もあまり重く感じることはないでしょう。MZ-80Kの半分の速度で動きます。

Miyajima Yasusi
宮島　靖

いやあ，最近学校関係で忙しかったせいか，なかなか記事を書けなくって昨年11月号のEDSACシミュレータ以来実に半年ぶりの登場なのでした。で，久しぶりに出てきてなにをやるかというと，Z80のシミュレータなんぞをやるわけです。

前回もEDSACという世界最初のノイマン型CPUのシミュレータをやったのですが，学習にはなるが実用性に乏しい（でも一部の大学では役に立ったはず）という欠点がありました。しかーし，今回はZ80のシミュレータであり，さらにS-OSまでも移植しちゃうよという，その筋の人にはなかなかにおいしい内容であるはずなのです。では具体的な内容を説明していきますが，「すごいじゃん」と思われたいので，難しいことも説明していきます。

で，各項目ごとにわかりやすい☆による5段階難易度をつけました。自分のレベル以上だと思う項目は，はしょっちゃってもかまわないし，背伸びしてファイブスターに挑戦するもよいでしょう。ちなみに，星の数とレベルの目安は次のとおり。

☆……………………猿でもわかる
☆☆………………初心者マークの人でもわかる
☆☆☆……………Oh! Xを読んでいる一般読者ならわかる
☆☆☆☆…………アセンブラが使える人ならわかる
☆☆☆☆☆………アセンブラが使いこなせる人ならわかる
☆☆☆☆☆☆……筆者にもわからない。よってこのレベルはなし

S-OS概略

●S-OSってなに？（☆☆）

X1やMZのユーザーだった人たちには，もうお馴染みだろうから読み飛ばしてもらってかまいません。が，新規X68000ユーザーにとってはS-OSはまったく知らないという人もいると思われるので，簡単に説明しましょう。

S-OSは最初はX1，MZなどのZ80というCPU用に開発されたプログラムで，簡単にいうと異機種間のハードウェアの差異を吸収してしまって，S-OSが動く機種ならどんな機種でも同じプログラムが動くようにするという優れものなわけです。いままでならX1用，PC-8801用とわざわざ2種類作らなきゃならなかったものがひとつ作るだけで両方で動いてしまうのです。まあ，こういうことをできるようにするためのものだと思ってください。

●ハードの差異吸収の原理（☆☆☆☆）

いきなり，4つ星で申し訳ない。でもこれを説明しないと中級以上の方にはもの足りないでしょうから。

ご存じのように，各メーカー，はたまた同一メーカー内でも機種の違いによって，画面の構成，VRAMアドレス，ディスク入出力など違うところだらけです。というか，同じ部分はCPUだけといったほうが正しいかもしれません。

でも考えてみると，CPUはちゃんとどの機種でも同じ動作をしているわけで，ただ単に周辺との入出力において違いが出ているだけなのです。じゃあ，そこをなんとかすればいいというわけで，考えられた方法が，OSによる低水準入出力の抽象化です。どの機種にも共通している機能の使用方法を同じにして，ユーザーの作成したプログラムでは直接ハードウェアの特定番地，特定I/Oポートをアクセスしないようにしてしまうのです。低水準の入出力をOSが行ってくれるのです。

OSはそういったサブルーチン群の塊であり，当然OS自体はその機種ごとにプログラムされていなければなりませんが，その上で動作するアプリケーションはすべて共通となります。S-OSでは，特定のレジスタにパラメータをセットして，特定の番地をコールすることによってOSのサブルーチンを呼び出す形式をとっています。

表4にS-OSの全ファンクションコールを載せてあります。これらのエントリアドレスは各機種用のS-OSで共通ですから，たとえば，画面に文字を表示したければ，どの機種でもAレジスタに文字コードを入れて，1FF4Hをコールすればよいのです。

●なぜ市販ソフトは機種専用のものが多いのか（☆☆☆）

では，市販のソフトはなぜそういう頭のいい方法をとらないのでしょうか。それにはれっきとした理由があります。まず，機種間でハードウェアの仕様がかなり異なる場合，それらを共通のOSで実現することが困難である。ハードウェアスクロールやPCGなんてPC-8801にはないし，FM音源だって全然違ったりなんかする。で，しょうがないからその部分はその機種専用で作ってしまうことになります。

次に，OSは汎用性を持たせるためと，安全性を持たせるために，いろいろなエラーチェックを行ったり，特定のプログラムには不必要とも思われることまでやってくれたりすることもあります。で，その結果速度が犠牲になってしまうことが多々あります。市販のゲームを作る場合には速度やその機種の性能をいかに引き出すかが重要なので（特に日本のゲームはこの傾向が強い。もっと本質的な面白さを追求してくれるとよいのだけど），結局その機種専用でプログラムしたほうがいろいろ自由なことができるというわけです。

まあ，あと実際には他機種の移植なんて最初からあんまり考えてなくって，とりあえずこの機種で作って，あとは移植する人に任せちゃおうかなっていう傾向もないわけじゃあないですね。

●S-OSのアプリケーション（☆☆）

S-OSはZ80のOSとして開発されて以来いろいろなソフトが発表されています。Z80のアセンブラのREDA，エディタのWINE

R, LISP, PROLOG, SLANGなどの言語も動いちゃうのです。これからもきっと，いろいろなソフトが出てくることと思われます。非常に簡単な説明でごめんなさい。だって，ついこの前までS-OSのことはまったく知らなかったから，アプリケーションについても詳しくは知らないんです。すみません。

X68000版“SWORD”

●**X68000版概略**（☆☆☆）

S-OSのメインマシンとX68000とではディスクなどのメディアがまったく違うため，2DのS-OSのディスクイメージをHumanのファイルに落として使用します。通常5インチ2Dは320Kバイトの容量なので，この大きさのファイルになります。

S-OSの伝統に則り，ディスクイメージファイルは4つまで指定できるようになっています（ドライブは4台)。

S-OS“SWORD”の起動はHuman68kのコマンドラインから，

SWORDディスクイメージファイル名

と入力すると，指定のイメージファイルを読み込んで立ち上がります。

表1～3はそれぞれモニタコマンド，キャラクタコード，“SWORD”エラーメッセージです。で，表1なのですが，X68000版は使用できるコマンドがひとつ増えています。Vコマンドがそれで，縦の行数の25行と31行切り換えを行います。ただ，ゲームなどでは，25行モードでないとおかしくなるものもあるので注意が必要です。

それから，M，！コマンドなのですが，X68000版は親プロセスに戻るようになっています。X68000にはマシン語モニタなるものはないので，こういう仕様になってます。まあ，どうしてもモニタに戻したいという場合には，デバッガ上で動かせば，似たようなもんですけど，あくまでもX68000上だということで，そのへんの知識がないとあんまり重箱の隅をつつくようなまねはできません。まあ，あとで上級者用に説明しますね。

●**ディスクイメージファイルの転送方法**（☆☆☆☆）

さて，問題はどのようにして2HDのディスクに2Dのディスクイメージを作るのかということです。いろいろな方法がありますが，筆者の行った方法を紹介しておきましょう。

まず，X1turboZのように2Dと2HDが扱える機械を用意して，2Dのセクタをひとつずつ読み込んで（BASICのDEVI$で簡単にできる)，2HDのセクタに連続して書いていきます（これもDEVO$で楽勝)。これを1280セクタ（16セクタ×80トラック）すべて書き出します。で，移した2HDはそのままでは Human では読めないので，自作のコンバートプログラムを作成してHumanのファイルにしてやります。セクタ長が1(つまり256バイトね)で書かれているのを読み込まなければならないので，IOCSコールの$46番のB_READを使用します。具体的には，リスト1のようにしてやると，A1レジスタのアドレスから320Kバイト読み込まれます。そのときに必ずこの領域はmallocなどで確保しておくことをお忘れなく。で，このバッファをDOSコールのwriteで書き出してやれば，Humanのファイルの出来上がり。

リスト1

```
moveq.l #$46,d0    * B_READ
move.w  #$9170,d1 * DRIVE 1,MFM
move.l  #$01_00_00_01,d2
* セクタ長 1,TRACK 0,SIDE 0,SECTOR 1
move.l  #327680,d3 * 320K Bytes
move.l  dskimg,a1
* malloc で確保した320Kのバッファアドレス
trap    #15
```

●**開発にあたって**（☆☆）

PC-8801版のソースを解析しながらしゃこしゃこと移植をしたんだけど，S-OSについて詳しくなかったから，移植している最中も「動くかな，ホントに」という不安で一杯でした。とりあえず本物のS-OSを見せてもらおうかなと思って，PC-8801用のものをコピーしたんだけど，編集部に忘れてきちゃうし，期限は迫るしで，結構あせって作ったんだけど，いやぁ，間に合ってよかった（なんじゃ，この文章は)。

●**技術的ハイレベルなお話**（☆☆☆☆☆）

さて，ちょっと突っ込んだ話をしましょう。まず，S-OSを動かすためにはCPUがZ80であるという条件が必要になるということはもちろん，それを68000CPUで実現するためにはZ80をソフト的にシミュレートしてやる必要があるということもファイブスターランクのあなたなら周知のことでしょう。もちろんそのとおりなのですが，Z80のシミュレータについては後述するのでそちらを見てください。

S-OSの本体は68000CPUで動かす以上，68000のアセンブラで記述しています。そのおかげで，ディスクのアクセスは8ビット機よりも高速であると思われます。速度的に不利なのはX68000のテキスト画面がビットマップであるため，1文字書くのに16バイト転送しなくてはならないという点で，一応高速化のために転送はmove命令を展開したり，アドレス計算もできるだけ速くするようにはしたつもりです。

2Dディスクの転送

X68000では2Dのディスクを読むことができません。メディアレベルでもかなり高い互換性を誇るS-OSシステムとしては非常に残念なことですが，X1turboで暫定的に用いられた“512Kバイト分だけ”サポートというのも悲しいので，メモリ上の320Kバイトを仮想2Dディスクとして扱うことにします。

PC-286シリーズを用いればデータ交換は可能ですが，ここではX1turboシリーズの2HDディスクを使ったファイル転送法を示します。

1) 2HDディスクをフォーマットする
2) 転送する2Dディスクを1ドライブに，2HDディスクを0ドライブに入れる
3) 下のディスクを実行
4) オマケディスクのSWORD.LZHを展開
5) TRANS.Xを実行

以上でHuman68kのディスク上に320K バイト分の仮想ディスクができあがります。もちろん，これはPC-286シリーズの仮想ディスクとしても使えます。

```
10 CLS:COLOR 5
20 PRINT "SWORD 2HD TRANSFER UTILITY Ver 1.00 Y.Miyajima"
30 COLOR 7:PRINT
40 PRINT "MOUNT 2D SWORD DISK ON DRIVE 0"
50 PRINT "MOUNT 2HD BUFFER DISK ON DRIVE 1"
60 COLOR 2:PRINT "HIT ANY KEY":COLOR 7
65 IF INKEY$="" THEN 65
70 DEVICE "0:0':DEVICE "1:2"
80 FOR SEC=0 TO 1279
82 LOCATE 5,10:PRINT USING "Now transfering #### record.",SEC
90     DEVI$ "0:",SEC,F1$,F2$
100    DEVO$ "1:",SEC,F1$,F2$
110 NEXT
120 PRINT "Wao!! Completed!! Wonderful!! Beautiful!! Wahaha!!"
```

が，やろうと思えばもっと速くすることもできます。フォントのアドレスの計算を前もってテーブルで持っておくとか，画面のXY座標（たったの3000個）からVRAMアドレスへ変換するのもテーブルを用いれば遅いシフト命令を用いずにすみます。まあ，とりあえずは現在のものでもそこそこ速いので，このままにしてあります。画面の消去とスクロールは，ラスタコピーを使用しているので，どの8ビット機よりも速いのではないでしょうかね。

S-OSのファンクションコールにはSCRNという画面上の指定座標に表示されているキャラクタをAレジスタに読み出すファンクションコールがあります。ビットマップには，そういうのがいちばん辛くて，仮想テキスト画面をメモリに持ってそこを読み出すという手法が一般的ですが，その方法だと画面のスクロール，消去時などに大量のバッファを操作しなくてはならず，ただでさえ遅い画面まわりがさらに重くなってしまいますし，結構管理が大変なので，もっと簡単で確実な方法を用いました。

S-OSは基本的に白黒テキストなので，VRAMは1枚ですみます。X68000のテキストは4画面あって，そのうちプレーン2と3は電卓，ソフトウェアキーボードに使用しており，電卓はとりわけ，おいしい機能なので残しておきたいから，プレーン0に文字を書いて，プレーン1の同じオフセットアドレスにキャラクタコードを書いておくことにしました。

これなら，高速にキャラクタコードを取り出せるし，スクロールや画面消去もハードが勝手にやってくれちゃうので，一石二鳥の手法です。

S-OSの呼び出しは前述のとおり，特定番地のコールで，それはZ80の命令によるものです。そこで，なんらかのかたちで68000で記述されたファンクションコールに制御を移してやらなければなりません。そこで，本物のZ80にはない命令を作ってやることにしました。

その命令はED_F7_nnという命令で，Z80では未定義の命令コードです。3バイト目のnnがファンクション番号を表していて，表4の＃COLDを0番として，順に1ずつインクリメントして，＃ERRORで$39番になっています。この疑似命令をそれぞれのファンクションコールのエントリアドレスに埋め込んでおけば，Z80のシミュレータがファンクションコールの命令と解釈して，適当にやってくれるという仕組みになっています。

Z80シミュレータ

●「マシンゴーZ80」について（☆）

去年の今頃，学校でZ80関係の授業があるということで，シミュレータを作るのが3度の飯の次の次くらいに大好きな僕としては，こりゃ，皆のために奮起してZ80開発ツールを作ってみるかと思い立ちました。仕様を決めるよりも先に，なんかゴロのいい名前が決まらないと製作に取り掛かる気が起きないたちなので「マジンガーZ」からヒントを得て「マシンゴーZ80」とすることにしたのです。

どうでもいいところに結構凝ってしまうたちなので，友人にZ'sSTAFFでマジンガーZとホバーパイルダーを描いてもらいました。それで，立ち上げたら「パイルダーオン！」とかなんとかPCMでしゃべらせようともくろんでいたのですが，結局途中で飽きてしまい完成させることができませんでした。いやぁ，みんなにあれほど宣伝しておいたのに……。

それ以来「やっぱ口だけ」とか，「7がけ」とかいわれて「なしかちゃ，みちょれよ」と思っていたら，ひょんなことからS-OSを動かしてみようという話がありまして，じゃあ，久しぶりにZ80シミュレータの続きでもやるかと思い立ったわけであります。で，今回完成させることができたのは，飽きる前に作りあげたという点につきますかね。やっぱ，大きなものを作るときは，集中して短期間にあげちゃうことが大切かもしれないなと。教訓ですねこれは。

●気になる速度は？（☆☆☆）

ソフトウェアでシミュレートしているわけですから，本物のZ80の速度なんて出るわけがないので，「16ビットなんだからX1よっかはえーだろうな」と考えてた人，あんたは甘ちゃん。

ちょっと前のバージョンは1MHz弱のスピードがあったんだけど，フラグの処理を以前より細かくしたので，現在のものは0.8MHzあるかないかってところだと思われます。今回はディスクに収録されていて打ち込む必要がないので各自で評価してみてください。

●内部仕様とか難しい話（☆☆☆☆☆）

ではレジスタの割り当てから，説明しましょう。ご存じのとおり，Z80にはA，B，C，D，E，H，L，Fとその裏レジスタおよびIX，IY，SP，PCがあります。これらを次のように割り当てました。

D2……上位A'，下位A
D3……上位BC'，下位BC
D4……上位DE'，下位DE
D5……上位HL'，下位HL
D6……上位IX，下位IY
D7……上位F'，下位F
A4……Z80のメモリの先頭番地
A5……SP
A6……PC＋A4

すべてレジスタに割り当てて，高速化を図っています。裏と表の交換もSWAP命令で行えるので便利です。頭を悩ませた問題は，A4レジスタの取り方とフラグでした。

まず，A4レジスタの取り方というのは，Z80のメモリの先頭番地をA4に与えるか，それともZ80のメモリ＋8000HをA4に与えるかということです。68000のアドレッシングを考えると前者より後者のほうが非常に都合がいいのです。たとえば，LD A,(HL)を実行したいなら，

前者

```
moveq.l #$00, d0
move.w  d5, d0
move.b  (a4, d0.l), d2
```

後者

```
move.b  (a4, d5.w), d2
```

と，後者のほうが8クロックも速く終了しますし，命令も短くてすむのです。

しかし，プログラムカウンタであるA6には，Z80のアドレスを入れるよりも68000の絶対アドレスで管理したほうが命令のフェッチが(A6)＋というアドレッシングで行えるので楽なのです。後者のように0番地が真ん中にあるような構造だと，7FFFHと8000H番地が68000のメモリ上ではつながっていないので（64Kバイトも離れている）8000H番地をまたぐプログラムは実行できなくなってしまいます。それを解決するためには，A6もZ80のメモリ番地にして，相対形式でフェッチを行うしかないわけです。そうすると，

```
move.b (A4, A6.w), d0
addq.w #1, A6
```

となり，フェッチにかかるクロック数が18クロックとなり，逆にトータルで2クロック遅くなってしまいます。しかも命令のフェッチは，毎回やらなければならない部分なので，この方式は却下することにして前者を採用しました。

次に，フラグ関係ですが，これが大変でした。いや，ほんとに。結局ハーフキャリ以外はすべてつけました。Nフラグなんていらないだろうと思っていたら，なぜかこれがないと動かないものがあったりして，困ったもんです。ハーフキャリは，やろ

うと思えばできないものではないですが，速度は極端に犠牲になるのを覚悟せざるをえないので，却下。まあ，DAAの10進補正なんてゲームのスコアくらいなもんでしょ，きっと。スコアが16進でもいーじゃん。見逃してくれよぉ。

そうそう，あと速度低下の原因のひとつに，インテル系特有の上位下位反転格納がありましたねぇ。さらに，68000ですから奇数番地からのワードアクセス禁止なんぞという掟が，なかなかどうして困りました。たとえばLD(nn),HLをやりたければ，

```
moveq.l  #$00,D0
move.w  (A6)+,D0
move.w  D5,(A4,D0.l)
```

ですみそうなもんですが，実際には，

```
moveq.l  #$00,D0
move.b  1(A6),D0
ror.w    #8,D0
move.b  (A6),D0
addq.l   #2,A6
move.b  D5,(A4,D0.l)
ror.w    #8,D5
move.b  D5,1(A4,D0.l)
ror.w    #8,D5
```

というふうにしないといかんのです。

68020からは，奇数番地からワードアクセスしても平気（ちょっと遅くなるけど）だから，68020なら4MHzのZ80と同等くらい(以上？)のものが作れそうなんだけどなぁ。

●問題点（☆☆☆☆）

まず，動かないソフトがありました。原因は究明しようと思っていますが，どの命令がバグっているのかなかなかわからないのです。今回のディスクに入っているものはバージョン0.9ということで，ソースプログラムもすべてついています。

で，お願いですが，もし「この命令が動作おかしいよ」ってな具合のものが見つかったらぜひ編集部まで，報告してください。バグは人海戦術で取りましょう。私ひとりが苦しむのはいやじゃ。まあ，これは冗談ですけど。でも，大勢の優秀な読者ならきっと，すぐにバグを発見してくれると期待してます。

次に，速度ですが，もう少し速くすることは可能です。画面表示，リアルタイムキースキャン，命令の実行部それぞれについて速くできる要素は残っていますので，暇を見つけてはバージョンアップを図りたいと思っています。

●最後に（☆）

全然コンピュータに関係ないんだけど，今年の学園祭にはぜひともアマバンで参加したいなと思ってるんだけど，ドラムがいないんですよ。やっぱり打ち込みじゃいやでしょ。生で叩かないと。で，三鷹とか小金井とかの付近でドラムやってる君！　一緒にやろうじゃないですか。あっそうそう，ぼくらの好んでいる音楽は，日本人ならZIGGYさんなのですが，洋楽のハノイやモトリーもやりたいと思っています。やってもいいぞっていう人は，「Oh!X　宮島さん，いっしょにやろうか」の係まで。ちなみにマジだよ。いい忘れたけど，当方ベースです。

あと3月号から記事を書き始めた鈴木氏と友人がギターやってます。バカうまとかじゃなくてかまわないですし(こっちもうまくないから，うますぎる人は恐縮してしまう)，完全プロ指向なしです。とりあえず，楽器っておもろいなぁ，バンドはもっとおもろいなぁ。という人は声かけてください。ここより「バンドやろうぜ」にでも書いたほうがよかったかな，こりゃ。失敬。

表1　X68000"SWORD"のモニタコマンド
（[ ]は省略可能であることを示す）

#D[〈デバイス名〉:]
　〈デバイス名〉で指定されたデバイスのディレクトリを表示する。省略時はデフォルトのディレクトリ。

#DV〈デバイス名〉:
　デフォルトデバイスを変更する。

#J〈アドレス〉
　アドレスから始まるプログラムをコールする。サブルーチン中のRETでS-OSのモニタにリターンできる。

#　〈ファイル名〉
　ファイルがバイナリファイルであればそのままロードして実行。ASCIIファイルであればバッチファイルとみなして実行する。

#K〈ファイル名〉
　〈ファイル名〉で与えられたファイルを消去する。

#L〈ファイル名〉[:〈ロードアドレス〉]
　〈ファイル名〉で与えられたファイルを〈ロードアドレス〉へロードする。ロードアドレスが省略されたときには，セーブしたときのアドレスへロードする。

#M
　親プロセスに戻る。

#N〈ファイル名1〉:〈ファイル名2〉
　〈ファイル名1〉を〈ファイル名2〉に変更する。なお，〈ファイル名2〉のデバイス指定は不要。

#S〈ファイル名〉:〈開始番地〉:〈終了番地〉[:〈実行番地〉]
　〈開始番地〉から〈終了番地〉までを〈ファイル名〉でセーブする。

#ST〈ファイル名〉:Pまたは:R
　〈ファイル名〉で指定されたファイルにライトプロテクトをかける。その後は同一ファイルのセーブ，消去ができなくなる。プロテクトをはずすにはRを指定。

#W
　画面の40字/80字モードを切り替える。

#V
　縦の行数25行/31行を切り替える。

#!
　親プロセスに戻る。

THE SENTINEL

表2　S-OSキャラクタコード

下位＼上位	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	nul		SP	0	@	P		p					タ	ミ		
1			!	1	A	Q	a	q			。	ア	チ	ム		
2			"	2	B	R	b	r			「	イ	ツ	メ		
3			#	3	C	S	c	s			」	ウ	テ	モ		
4			S	4	D	T	d	t				エ	ト	ヤ		
5			%	5	E	U	e	u				オ	ナ	ユ		
6			&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7			'	7	G	W	g	w			ァ	キ	ヌ	ラ		
8			(	8	H	X	h	x			ィ	ク	ネ	リ		
9			)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A			*	:	J	Z	j	z			ェ	コ	ハ	レ		
B		BRK	+	;	K	[	k	■			ォ	サ	ヒ	ロ		
C		→	,	<	L		l				ャ	シ	フ	ワ		
D	CLS	←	−	=	M	]	m				ュ	ス	ヘ	ン		
E	CR	↑	.	>	N	^	n				ョ	セ	ホ	゛		
F		↓	/	?	O		o	π			ッ	ソ	マ	゜		

表3　エラーメッセージ

No.	メッセージ	内　容
1	Device I/O Error	入力時にエラーが発生した
2	Device Offline	デバイスがつながっていない
3	Bad File Descriptor	ファイルディスクリプタが間違っている
4	Write Protected	ライトプロテクトがかかっている
5	Bad Record	レコードナンバーに間違いがある
6	Bad File Mode	アトリビュートが違う
7	Bad Allocation Table	ファットエラー
8	File not Found	ファイルが見つからない
9	Device Full	ディスクがいっぱい
10	File Already Exists	すでに同名のファイルが登録されている
11	Reseved Feature	現在使用されていない
12	File not Open	ファイルをオープンせずに読み書きしようとした
13	Syntax Error	文法間違い
14	Bad Data	正しい引数ではない

表4 S-OSのサブルーチン一覧表

ルーチン名 (アドレス)	サブルーチンの機能	レジスタ破壊
#COLD (1FFD$_H$)	S-OSのコールドスタート。初期設定後メッセージを出力し，ワークエリア#USRに格納されているアドレスにジャンプする。#USRには初期値として#HOTアドレスが格納されている。	―
#HOT (1FFA$_H$)	S-OSのモニタになっており，プロンプト#が出てコマンド入力待ちになる。	―
#VER (1FF7$_H$)	HLレジスタにS-OSの機種とバージョンを返す。Hレジスタは機種を表しており，上位4ビットで機種の系列を示し，下位4ビットで系列内の機種番号を示す。 0 0 MZ-80K/C/1200 0 1 MZ-700 0 2 MZ-1500 1 0 MZ-80B 1 1 MZ-2000/2200 1 2 MZ-2500 2 0 X1 2 1 X1turbo 3 0 PC-8801(ROM) 3 1 PC-8001 3 2 PC-8801(オールRAM) 4 0 FM-7/77 5 0 SMC-777 6 0 PASOPIA 6 1 PASOPIA7 7 0 PC-286 8 0 X68000 LレジスタはS-OSのバージョンNo.を示しており，各種パッケージを追加したりした場合のS-OSのバージョンをチェックできるようにする。	HL
#PRINT (1FF4$_H$)	Aレジスタの内容をASCIIコードとみなし表示する（1文字表示）。	F
#PRNTS (1FF1$_H$)	スペースをひとつ表示する。	F
#LTNL (1FEE$_H$)	改行する。	なし
#NL (1FEB$_H$)	カーソルが行の先頭になければ改行する。	なし
#MSG (1FE8$_H$)	DEレジスタの示すアドレスから0D$_H$があるまでASCIIコードとみなし文字列表示する。	F
#MSX (1FE5$_H$)	DEレジスタの示すアドレスから00$_H$があるまでASCIIコードとみなし文字列表示する。	F
#MPRNT (1FE2$_H$)	これをコールした次のアドレスから00$_H$があるまでASCIIコードとみなし文字列表示する。 例） CALL #MPRNT DM "MESSAGE" DB 0	AF DE
#TAB (1FDF$_H$)	Bレジスタの値とカーソルX座標との差だけスペースを表示する。	AF
#LPRNT (1FDC$_H$)	Aレジスタの内容をASCIIコードとみなしプリンタのみに出力する。プリンタエラーがあった場合は，キャリフラグをセットしてリターンする。	AF
#LPTON (1FD9$_H$)	上記#PRINT～#TAB，#PRTHX，#PRTHLの出力をディスプレイだけでなくプリンタにも出力するかどうかのフラグ#LPTSWをセットする。これをコールしたあとは，上記サブルーチンでプリンタにも出力される。	なし
#LPTOF (1FD6$_H$)	フラグ#LPTSWをリセットする。これをコールしたあとは，#PRINT～#TAB，#PRTHX，#PRTHLの出力をディスプレイのみにする。	なし
#GETL (1FD3$_H$)	DEレジスタにキー入力バッファの先頭アドレスを入れてコールすると，キーボードから1行入力をして文字列をバッファに格納しリターンする。エンドコードは00$_H$。途中でSHIFT＋BREAKが押されたら，バッファ先頭に1B$_H$が格納される。	AF
#GETKY (1FD0$_H$)	キーボードからリアルタイムキー入力をする。入力したデータはAレジスタに格納され，何も押されていないときはAレジスタに0をセットしてリターンする。	AF
#BRKEY (1FCD$_H$)	ブレイクキーが押されているかどうかをチェックする。押されているときはゼロフラグをセットしてリターンする。	AF
#INKEY (1FCA$_H$)	なにかキーを押すまでキー入力待ちをし，キー入力があるとリターンする。押されたキーのASCIIコードはAレジスタにセットされる。	AF
#PAUSE (1FC7$_H$)	スペースが押されていれば，再び何かキーを押すまでリターンしない。このときSHIFT＋BREAKを押すと，このルーチンをコールした次のアドレスの2バイトの内容を参照し，そこへジャンプする。 例） CALL #PAUSE DW BRKJOB ここでブレイクキーを押すとBRKJOBへジャンプ， さもなくばDW BRKJOBはスキップ。	AF
#BELL (1FC4$_H$)	ベル（ピープ音）を鳴らす。	AF
#PRTHX (1FC1$_H$)	Aレジスタの内容を16進数2桁で表示する。	AF
#PRTHL (1FBE$_H$)	HLレジスタの内容を16進数4桁で表示する。	AF
#ASC (1FBB$_H$)	Aレジスタの下位4ビットの値を16進数を表すASCIIコードに変換し，Aレジスタにセットする。	AF
#HEX (1FB8$_H$)	Aレジスタの内容を16進数を表すASCIIコードとしてバイナリに変換し，Aレジスタにセットする。Aレジスタの内容が16進数を表すASCIIコードでない場合は，キャリフラグをセットしてリターンする。	AF
#2HEX (1FB5$_H$)	DEレジスタの示すアドレスから2バイトの内容を，2桁の16進数を表すASCIIコードとしてバイナリに変換し，Aレジスタにセットする。エラーの場合はキャリフラグがセットされる。	AF DE(+2)
#HLHEX (1FB2$_H$)	DEレジスタの示すアドレスから4バイトの内容を，4桁の16進数を表すASCIIコードとしてバイナリに変換し，HLレジスタにセットする。エラーがあった場合は，キャリフラグがセットされる。	AF HL DE(+4)
#WOPEN (1FAF$_H$)	#FILEでセットされたファイル名，(#DTADR)，(#SIZE)，(#EXADR)をテープに書き込む。ディスクの場合は，新しいファイルかどうかのチェックを行う。エラー発生時にはキャリフラグが立つ。	AF BC DE HL
#WRD (1FAC$_H$)	(#DTADR)，(#SIZE)，(#EXADR)に従って，デバイスにデータをセーブする。ディスクの場合#WOPEN後でないとFile not Openのエラーが出る。	〃
#FCB (1FA9$_H$)	テープの場合"MACE"の#RDIとまったく同じ。ディスクの場合#DIRNOの値に従って（#IBFAD）にディレクトリの内容を転送する。これにより"MACE"用プログラムにまったく手を加えることなくディスクリードを行うことができる。CALL後，(#DIRNO)はインクリメントされる。ブレイクキーが押されると(#DIRNO)をクリアする。リターンキーが押されるとキャリフラグを立ててリターンする。	〃
#RDD (1FA6$_H$)	(#DTADR)，(#SIZE)，(#EXADR)に従って，デバイス上のファイルを読み込む。#ROPEN後でないとFile not Openのエラーが出る。	〃
#FILE (1FA3$_H$)	Aレジスタにファイルのアトリビュート，DEレジスタにファイル名の入っている先頭アドレスをセットしてコールすると（#IBFAD）にファイル名のセットと（#DSK）にファイルディスクリプタのセットを行う。ファイルを操作する前には，必ずこのサブルーチンにより，ファイル名とアトリビュートをセットしなければならない。コール後DEレジスタは行の終わり（00$_H$）か：(コロン）の位置を示している。	〃
#FSAME (1FA0$_H$)	#FILEでセットされたファイルネームと，読み込んだファイルネームを比較する。一致すればゼロ，不一致ならばノンゼロでリターンする。アトリビュートのチェックも同時に行う。	〃
#FPRNT (1F9D$_H$)	デバイスから読み込んだファイルネームを表示する。スペースキーを押すと表示後一時停止する。	〃
#POKE (1F9A$_H$)	HLレジスタの内容をオフセットアドレスとして，S-OS用特殊ワークエリアにAレジスタの内容を書き込む。	なし
#POKE@ (1F97$_H$)	メインメモリからS-OS用特殊ワークエリアにデータを転送する。HLレジスタにメモリ先頭アドレス，DEレジスタにワークエリアオフセットアドレス，BCレジスタにバイト数を入れてコールする。	AF BC DE HL
#PEEK (1F94$_H$)	HLレジスタの内容をオフセットアドレスとして，S-OS用特殊ワークエリアからAレジスタにデータを読み出す。#POKEと逆の動作。	AF
#PEEK@ (1F91$_H$)	S-OS用特殊ワークエリアからメインメモリにデータを転送する。HL，DE，BCレジスタにセットするパラメータは#POKE@と同じ。	AF DE BC HL

#MON (1F8E$_H$)	各機種のモニタにジャンプする。	—
[HL] (1F81$_H$)	HLレジスタにコールしたいアドレスを入れ， CALL [HL] と使うことにより，擬似的なレジスタ間接コールが可能。	なし
#GETPC (1F80$_H$)	現在のプログラムカウンタの値をHLにコピーする。	HL
#DRDSB (2000$_H$)	DEが示すレコードナンバーからAが示すレコード数だけHLが示すアドレスに読み込む。連続セクタリード。(#DSK)にデバイス(A～D)をセットしてコールする。 LD DE，(#FATPOS) LD HL，(#FATBF) LD A，I CALL #DRDSB とすれば，FATバッファにFATを読み出すことができる。	AF AF′
#DWTSB (2003$_H$)	HLが示すアドレスからAレコード分(A×256バイト)の内容を，DEを先頭レコードとして記録する。連続セクタライト。(#DSK)にデバイス(A～D)をセットしてコール。	AF AF′
#DIR (2006$_H$)	(#DSK)で指定されたデバイス上の全ディレクトリを表示する。	AF BC DE HL
#ROPEN (2009$_H$)	テープの場合は，先に#FILEでセットされたファイル名と，読み込んだIBを比較し，同一ファイルならゼロ，違えばノンゼロでリターンする。ディスクの場合は，#FILEでセットされたファイルがディスク上にあるかどうかのチェックを行う。ゼロフラグは常にリセットとなる。いずれの場合にも，エラーが発生したときにはキャリでリターンする。またファイルの情報は，(#DTADR)，(#SIZE)，(#EXADR)へ転送される。	〃
#SET (200C$_H$)	#IBFADで示されるIBバッファの内容と一致するディスク上のファイルをライトプロテクトする。	〃
#RESET (200F$_H$)	#IBFADで示されるIBバッファの内容と一致するファイルのプロテクトをはずす。	〃
#NAME (2012$_H$)	#FILEで設定されたファイル名を，DEレジスタが示すメモリ上のデータに変える。リネーム。メモリ上のデータ中にデバイスディスクリプタが入っていても無視する。またDE+16以内にエンドコード(00$_H$，':')がないときにはエラーが発生する。	〃
#KILL (2015$_H$)	#IBFADで示されるIBバッファの内容と一致するディスク上のファイルを削除する。	〃
#CSR (2018$_H$)	現在のカーソル位置を，HにY座標，LにX座標の順で読み出す。カーソル位置の読み出しは必ずこの方法によること。(#XYADR)は使わない。	HL
#SCRN (201B$_H$)	HにY座標，LにX座標をセットしコールすると，画面上の同位置にあるキャラクタをAに読み出す。	AF
#LOC (201E$_H$)	HにY座標，LにX座標を入れてコールすると，カーソル位置がそこにセットされる。カーソル位置の設定は必ずこの方法によること。	AF
#FLGET (2021$_H$)	カーソル位置で，カーソル点滅1文字入力を行い，Aに押されたキャラクタをセット。オートリピートもかかる(MZ-80K/C/1200は不可)。画面へのエコーバックは行わない。	AF
#RDVSW (2024$_H$)	デフォルトデバイスをAに読み出す。デフォルトデバイスを知りたいときには必ずこの方法によるものとする。	A
#SDVSW (2027$_H$)	デフォルトにしたいデバイス名をAに入れコールすると，デフォルトデバイスがセットされる。必ずこの方法によること。(#DVSW)を直接触ることを禁止する。	AF
#INP (202A$_H$)	共通I/OポートからlバイトをAに読み込む。ポートはCで指定する。	AF
#OUT (202D$_H$)	共通I/OポートへAを出力する。ポートはCで指定する。	なし
#WIDCH (2030$_H$)	画面のモード(40字，80字)を切り換える。Aに40以下の数をセットすると40字，40より大きい数をセットしてコールすると80字となる。現在のモードは(#WIDTH)に入っている。この機能はMZ-80K/C/1200/700/1500にはない。	AF BC DE HL
#ERROR (2033$_H$)	Aにエラー番号をセットしてコールすることによりエラーメッセージを表示する。	〃

表5 S-OSのワークエリア

ワーク名（アドレス，バイト数）	内　　容
#USR (1F7E$_H$～，2バイト)	S-OSをコールドスタートしたあとジャンプするアドレスを示している。通常はS-OSのホットスタートのアドレスになっている。
#DVSW (1F7D$_H$，1バイト)	テープフォーマットなどを切り換えるフラグ。 0：MZフォーマット2400ボー(共通モード) 1：各機種のモニタに依存 3：QD(MZ-1500のみ) コールドスタート時は0になっている。
#LPSW (1F7C$_H$，1バイト)	#PRINT～#TAB，#PRTHX，#PRTHLルーチンでの出力をディスプレイだけでなくプリンタにも出力するかどうかのフラグ。0以外でプリンタにも出力。コールドスタート時は0になっている。
#PRCNT (1F7A$_H$～，2バイト)	改行してから表示した文字数を格納してあるアドレスを示している。
#XYADR (1F78$_H$～，2バイト)	カーソル座標が格納されているアドレスを示している。
#KBFAD (1F76$_H$～，2バイト)	各機種のキー入力用バッファのアドレスを示している。 例) LD DE，(#KBFAD) CALL #GETL
#IBFAD (1F74$_H$～，2バイト)	インフォメーションブロックの先頭アドレスを示している。同時にファイルアトリビュートのアドレスでもある。
#SIZE (1F72$_H$～，2バイト)	ファイルサイズ。#WOPEN，#WRD，#FCB，#RDD，#ROPENルーチンで使用される。
#DTADR (1F70$_H$～，2バイト)	ファイル先頭アドレス。
#EXADR (1F6E$_H$～，2バイト)	ファイルのエントリアドレス。
#STKAD (1F6C$_H$～，2バイト)	各機種のモニタが使用しているスタックのアドレスを示している。
#MEMAX (1F6A$_H$～，2バイト)	S-OSで使用できるメモリの上限を表す。
#WKSIZ (1F68$_H$～，2バイト)	S-OS用特殊ワークエリアのサイズを表す。
#DIRNO (1F67$_H$，1バイト)	#FCBで使用するワーク。このワークに値を入れて#FCBをコールすると，先頭から数えてその値で示されるFCBを(#IBFAD)にロードする。ロード後，値は1増える。
#MXTRK (1F66$_H$，1バイト)	使用できる最大トラック数が入っている。
#DTBUF (1F64$_H$～，2バイト)	ディスクからデータを読み込む先頭アドレスが入っている。データバッファは256バイト。
#FATBF (1F62$_H$～，2バイト)	ディスクからFATを読み込む先頭アドレスが入っている。FATバッファは256バイト。
#DIRPS (1F60$_H$～，2バイト)	ディレクトリが入っているレコードナンバーの始まりを示す。S-OS"SWORD"では10$_H$。書き換えることによってディレクトリの位置を移動できる。
#FATPOS (1F5E$_H$～，2バイト)	ファイルアロケーションテーブル(FAT)が入っているレコードナンバーを示す。S-OS"SWORD"では0E$_H$。書き換えることによりFATの位置を移動することができる。
#DSK (1F5D$_H$，1バイト)	アクセスしようとするデバイス名が入る。
#WIDTH (1F5C$_H$，1バイト)	現在のスクリーンモードが入っている。 40字の場合：28$_H$ 80字の場合：50$_H$ MZ-80K/C/1200/700/1500は横40字固定。
#MAXLN (1F5B$_H$，1バイト)	画面に表示できる最大行数が入っている。

特別付録

PC-286/9801用S-OS“SWORD”

Z80, 6809, 68000に続く全機種共通システムの8086版。ついにPC-286用のS-OSの登場です。これで日本パソコンの過半数で同じプログラムが走ることになったわけです。紙面と磁性面の都合によりソースは掲載できませんでした。ごめんなさい。

Endo Takashi
遠藤 隆

お待たせしました。ついにS-OSがPC-286上でも動くようになりました。PC-286のCPUは8086シリーズです。当然，Z80の命令は実行できません。そこで，ソフトでZ80をエミュレートさせています。最近のPC-286およびPC-9801シリーズは80386や80286を積んでいて，ソフトウェアエミュレートでもそこそこの速さが出ます(12MHzの80286で1MHzのZ80相当)。

S-OSサブルーチンの処理はDOSモジュールに用意されているものはそちらを呼び，I/O関係のDOSモジュールにはないものは8086で記述してあります。Z80エミュレートから8086への制御の切り替えは，Z80のHALT命令で行っています。Z80エミュレータがHALT (76H) にぶつかるとその次の1バイトを取り出し，それに従って必要な8086ルーチンを呼び出すかたちになっています。

また，ディスクメディアは一応，2Dディスクが読めるようにしてありますが，これは2Dからのファイルの転送目的のためのおまけの機能です。通常使う場合にはMS-DOS上に320Kのファイルを作り，それを仮想ドライブとして使います。

そのため，このS-OSを使うにはMS-DOS(3.1以上)が必要です。また，MS-DOSが立ち上がっている状態で180Kバイト近くのフリーエリアが必要です。最近のマシンはすべて，最初からメモリを640K以上積んでいるので問題はないと思います。

“SWORD”用システムディスクの作成

さて，いよいよSWORD用のシステムディスクの作成です。まず，ディスクを用意します。そのディスクをMS-DOSでフォーマットしてシステムを書き込んでください。

そしてそのディスクにSWORD.EXEをコピーして，次の内容のAUTOEXEC.BATファイルを作って書き込んでください。

sword

これでS-OSのシステムディスクの完成です。好みに応じてキャッシュディスクドライバなどを組み込んでください。仮想ディスク装置へのアクセスが速くなります。また，SWORDは漢字をサポートしていないので，FEPは組み込んでも無意味です（ハードディスクにSWORDを載せることももちろん可能です。MS-DOSの知識をフルに活用して使いやすい環境を構築してください)。

それでは作ったシステムディスクを立ち上げてみましょう。立ち上がるとタイトルが表示されるはずです。まだ仮想ディスクファイルがありませんので，まずは仮想ディスクファイルを作ってやらなければなりません。

そこで，なにもいわずに次のような操作をしてください。下線があなたの入力する部分です。

#M↵	(モニタに入る)
:FA↵	(デバイスA用の仮想ディスクファイル作成)
:!↵	(SWORDに戻る)
#	(もしここで，「?」と表示されたら，なんらかの原因，たとえばディスクの容量が足りなかったなどにより，仮想ディスクファイルの作成に失敗したことを示します)

これでデバイスA用の仮想ディスクファイルができます。一度仮想ディスクファイルを作成すれば，次からはこの操作は不要です。

“SWORD”の使い方

さて，これで準備がすべて整いました。それではSWORDの使い方です。DOSモジュールを使っているので，使い方は他機種版と基本的に変わりません。ただブートコマンドの！はSWORDを終了し，MS-DOSに戻るコマンドになっています。また，Mコマンドではこのあと紹介する独自の内部コマンドに飛ぶようになっています。

DOSモジュールは1989年2月号に掲載されたX1版のSWORDのものにパッチを当てて使っているので，バッチ処理などの拡張機能はサポートされていません。

RAMディスクの機能は最初からサポートしています。このRAMディスク用のメモリはZ80とはまったく別のメモリ空間にあるので，そのために特殊ワークが侵食されることはありません。ちなみにPC-286版SWORDでは特殊ワークエリアは64Kバイト確保されています。

デバイス名の割り当ては，

A～D	MS-DOS上の仮想ディスクファイル
E	本体RAMを用いたRAMディスク
F	リザーブ
G, H	ドライブ#1, #2を2Dディスクとしてアクセス

となっています。ここでA～Dの仮想ディスクファイルのファイル名はデフォルトでは“A:¥SWORD.DSK”～“D:¥SWORD.DSK”となっていますが，SWORDを起動するときに別のファイル名を指定できます。書式は次のとおりです。

SWORD [<Aのファイル名> [<Bのファイル名> [<Cのファイル名> [<Dのファイル名>]]]]

この書式に従って，AUTOEXEC.BATの内容を書き換えてください。たとえば，AUTOEXEC.BATの内容を，

SWORD a:¥s－os¥sword1.dsk a:¥s－os¥sword2.dsk

とすると，

Aにはa:¥s－os¥sword1.dsk
Bにはa:¥s－os¥sword2.dsk
Cにはc:¥sword.dsk(デフォルト値)
Dにはd:¥sword.dsk(デフォルト値)

が割り当てられます。

また，EのRAMディスクは起動時に320

Kバイト以上RAMが余っていた場合，それをRAMディスクとして割り当てて使います。RAMディスクが使えるときはその旨が起動時にメッセージとして表示されます。

それからG，Hはドライブ1，2に入っている2Dディスクを読み書きするときのデバイス名です。たとえば，2Dディスクをドライブ1に入れて，デバイス名Gでアクセスしてください。

エミュレータについて

本エミュレータでは未定義命令はまったくサポートしていません。未定義命令はすべてNOPとして処理されます。また，I/Oポートを操作する命令はX1のG-RAMへのアクセスだけはエミュレートしていますが，それ以外はNOPとして処理されます。このため，X1のMAGICが無改造で動作します。

LD A,Rについてはこれを乱数発生器として使っているプログラムもあるため，ある領域のROMデータを順次拾ってくることによってエミュレートしています。

速度最優先のため，1命令ごとに（！）処理ルーチンを作り，各命令の処理ルーチンのアドレスのテーブルを用意してZ80命令をフェッチしたらテーブルに従って各命令の処理ルーチンにダイレクトにジャンプしています。しかも各命令ごとに処理ルーチンの後ろにフェッチ&ジャンプ処理があるので（ジャンプ命令の時間すら惜しい）似たようなルーチンがだらだらと続き，プログラムが異常に長くなってしまいました。16ビットCPUだからこそできる，大技といえましょう。メモリが許す限りめちゃくちゃをやればもうちょっとは速くなりますが，もともとリスト掲載のことを考慮したうえで開発していましたので，このあたりで止めておきました。

Z80から見たときのメモリマップは図1のようになります。このZ80用のアドレス空間はデータセグメントでアクセスされています。また，特殊ワークエリアはこれとは別のアドレス空間に64Kバイト確保されています。

ここで注意してほしいのはZ80のレジスタがZ80のメモリ上にマッピングされているということです。はっきりいって超危険ですが，速度最優先なので（セグメントオーバーライトプリフェックスの時間すら惜しい）しかたありません。スタックポインタやプログラムカウンタはエミュレート中はレジスタに置かれているので，よっぽど変なことをしなければそれが原因でぶっとんじゃうことはないと思います。

図1

アドレス	領域
0000	Z80割り込みベクトル
0080	インフォメーションブロック
00A0	Z80レジスタ用ワーク 下手にいじるとぶっとぶぞー
00BC	BIOS用ワークエリア
04D8	Z80スタックエリア
120E	空きエリア
1F5B	S-OSワークエリア
2100	DOSモジュール
3000	ユーザエリア
FFFF	

そのほかの注意事項

本SWORDではS-OSサブルーチン群中の#IPNと#OUTの2つはサポートしていません。SWORDシステム中ではSTOPキー割り込みのベクタを横取りしていて，SWORDを使用中にSTOPキーによるハードディスクのシッピングはできません。STOPキーでシッピングしたい場合は#！でM

THE SENTINEL

専用モニタの使い方

モニタはSWORD上から，

#M.↵

とすると起動できます。このモニタは

1）メモリエディット
2）仮想ディスクファイルの作成
3）リアルタイムキー入力モード設定

の3つの機能しかない簡単なモニタです。

モニタを起動するとプロンプトが“：”に変わり，モニタモードになっていることを示します。それではモニタの各機能について説明していきましょう（[ ]は省略可能）。

:E [<アドレス> [<アドレスの基底> [<終了アドレス>]]]

メモリエディット

この機能はメモリの内容を書き換えるための機能です。エディットは128バイトのブロックを基本単位として行い，そのブロック内でカーソルキーと16進キー（テンキーの回りのキーが16進キーとして使えます）を使ってエディットを行います。次のブロックをエディットしたいときは「ROLL UP」キー，前のブロックをエディットしたいときは「ROLL DOWN」キーを押してください。また，エディットを終了するときは「ESC」キーです。縦，横のチェックサムおよびCRCチェックバイトは入力とともに逐次計算しなおされます。

アドレスの基底とは，たとえば128バイトのブロックの始まりの下位バイトが00Hや80H以外の中途半端なところから始まっている場合にブロックの始まりを??00や??80以外の値に指定するときに使います。当然，その性質上下位7ビット分しか意味を持ちません。たとえば，3055Hから始まるブロックをエディットしたい場合は，

:E 3055 55.↵

とします。

終了アドレスとはその名のとおり，ダンプリストの終了のアドレスです。これは128バイトのブロックの途中でダンプリストが終わってしまうときにその終了アドレスを指定するのに使います。デフォルトはFFFFHです。終了アドレスよりも後ろにあるメモリエリアはエディットすることができませんので気をつけてください。

たとえば，3000Hから始まり3577Hで終わるダンプリストをエディットしたい場合は，

:E 3000 0 3577.↵

とします。

この終了アドレスはエディット画面の左下に常に表示されています（下の図の例ではFFFFH）。その上に表示されている値（下の図の例では3015H）は現在のカーソルの位置のアドレスです。

```
3000:53 57 4F 52 44 00 00 00:8F SWORD...
3008:00 00 00 00 00 00 00 00:00 ........
3010:00 00 00 00 00 00 00 00:00 ........
3018:00 00 00 00 00 00 00 00:00 ........
3020:00 00 00 00 00 00 00 00:00 ........
3028:00 00 00 00 00 00 00 00:00 ........
3030:00 00 00 00 00 00 00 00:00 ........
3038:00 00 00 00 00 00 00 00:00 ........
3040:00 00 00 00 00 00 00 00:00 ........
3048:00 00 00 00 00 00 00 00:00 ........
3050:00 00 00 00 00 00 00 00:00 ........
3058:00 00 00 00 00 00 00 00:00 ........
3060:00 00 00 00 00 00 00 00:00 ........
3068:00 00 00 00 00 00 00 00:00 ........
3070:00 00 00 00 00 00 00 00:00 ........
3078:00 00 00 00 00 00 00 00:00 ........
-----------------------------------------
3015 53 57 4F 52 44 00 00 00 CRC  SUM
FFFF                         98F5 018F
```

また，「TAB」キーを押すことにより，16進キー入力と文字入力が切り替えられます。文字入力になるとカーソルは右側に移り，入力した文字のASCIIコードが入力されます。

:G

リアルタイムキー入力モード設定

この機能はリアルタイムキー入力（#GETKY）をBIOSを使って行うか，プログラムから直接行うかを指定します。BIOSを用いるモードをUtilityモード，プログラムから直接行うのをGameモードと呼んでいます。トグルになっていて，この機能を呼び出すごとにUtilityモードとGameモードが入れ替わります。なぜこのように2つのモードがあるかというと，BIOSを使ったキー入力（Utilityモード）ではゲームのときに操作がぎくしゃくした感じになってしまうし（ものによってはキー操作を受け付けない），プログラムから直接行う（GAMEモード）とユーティリティを使うときにキーリピートが速すぎて使いにくくなる場合があるためです。起動時は「Gameモード」になっています。ほとんどの場合はこのままで不都合はないはずです。

また，プログラムを実行中に不都合を感じたら，「F1」キーを押してください。これによってもキー入力モードを切り替えられます。

:F [<デバイス名>：]

仮想ディスクファイルフォーマット

この機能は仮想ディスクファイルを作るためのものです。デバイス名を省略するとデフォルトデバイスが選ばれます。また，すでに仮想ディスクファイルが存在する場合は，

already　formatted

と表示して，なにも行いません。ディスクの空き容量が足りなかったり，ディスクが挿入されていないと，

?

と表示されます。仮想ディスクファイルを消したい場合は，MS-DOS上でDELコマンドで消してください。

また，デバイス名にG，Hを指定すると，ドライブ1，2に入っているディスクを2Dに物理フォーマットします。

:!

モニタ終了

モニタを抜けて，SWORDに戻ります。

S-OSのキャラクタコードとPC-286のASCIIコードは異なる部分がままありますが，内部に変換テーブルを用意してS-OSのキャラクタコードに合わせられています。

したがって，キーボードから“|”と入力すると画面上では“■”と表示されます。また，同様に“¥”と入力しても“ ”と表示されます。このほか“π”は98にはないので，代わりに“円”と表示しています。S-OSのキャラクタコードについては125ページの表2を参照してください。

S-DOSに戻ってから行ってください。

デバイス名G，Hへの操作はPC-286では問題ないと思いますが，2HD，2DD両用ディスクを積んでいない機種については動作確認していないので，もしかすると2Dディスクへのアクセスの機能は使えないかもしれません。#GETL(1行入力)ではインサートキーが使えます。インサートキーを押すたびに入力モードがインサートとオーバーライトの間で入れ替わります。インサートモードのときはカーソルが小さくなるので，いまどのモードなのかが識別できます。

＊

これでPC-286，もしくはPC-9801をお持ちのあなたもすばらしいS-OSの世界に踏み込めます。まだPC-8001やMZ-80K/Cが覇を競っていた頃の，パソコンに対するあの純粋な情熱がS-OSによって再び蘇ってくるはずです。さあ，あなたも今日からS-OSを始めてみませんか？

謝辞

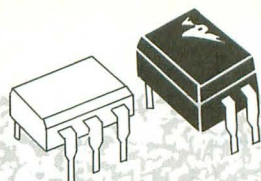
今回のプログラムの作成にあたり，多大なる助力をしてくださったDr.川手さん，遅いとかCで書き直してSunで走らせろとか無茶ばかりいったChar.坂口さんに感謝いたします。

参考文献

アスキー出版テクライト，「PC-9800シリーズテクニカルデータブック」，アスキー，1988
「X1版S-OS"SWOSD"」，Oh！X 1989年2月号
堀内保秀・木越聖，「FM-7/77版S-OS"SWORD"」，Oh！MZ 1987年8月号
庄司渉・本田稔，「Z80マイコンプログラミングテクニック」，電波新聞社，1981
近藤環，「マルチウィンドウエディタWINER」，Oh！X 1988年8月号
石神留二，「魔術師への道」，Oh！MZ 1986年9月号

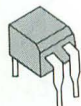
全機種共通システムインデックス

■85年6月号
序論　共通化の試み
第1部　S-OS"MACE"
第2部　Lisp-85インタプリタ
第3部　チェックサムプログラム
■85年7月号
第4部　マシン語プログラム開発入門
第5部　エディタアセンブラZEDA
第6部　デバッグツールZAID
■85年8月号
第7部　ゲーム開発パッケージBEMS
第8部　ソースジェネレータZING
■85年9月号
インタラプト　S-OS番外地
第9部　マシン語入力ツールMACINTO-S
第10部　Lisp-85入門(1)
■85年10月号
第11部　仮想マシンCAP-X85
連載　Lisp-85入門(2)
■85年11月号
連載　Lisp-85入門(3)
■85年12月号
第12部　Prolog-85発表
■86年1月号
第13部　リロケータブルのお話
第14部　FM音源サウンドエディタ
■86年2月号
第15部　S-OS"SWORD"
第16部　Prolog-85入門(1)
■86年3月号
第17部　magiFORTH発表
連載　Prolog-85入門(2)
■86年4月号
第18部　思考ゲームJEWEL
第19部　LIFE GAME
連載　基礎からのmagiFORTH
連載　Prolog-85入門(3)
■86年5月号
第20部　スクリーンエディタE-MATE
連載　実戦演習magiFORTH
■86年6月号
第21部　Z80TRACER
第22部　magiFORTH TRACER
第23部　ディスクダンプ&エディタ
第24部　"SWORD" 2000 QD
連載　対話で学ぶ magiFORTH
特別付録　PC-8801版S-OS"SWORD"
■86年7月号
第25部　FM音源ミュージックシステム
付録　FM音源ボードの製作
連載　計算力アップのmagiFORTH
特別付録　SMC-777版S-OS"SWORD"
■86年8月号
第26部　対局五目並べ
第27部　MZ-2500版S-OS"SWORD"
■86年9月号
第28部　FuzzyBASIC発表
連載　明日に向かって magiFORTH
■86年10月号
第29部　ちょっと便利な拡張プログラム
第30部　ディスクモニタDREAM
第31部　FuzzyBASIC 料理法〈1〉
■86年11月号
第32部　パズルゲーム HOTTAN
第33部　MAZE in MAZE
連載　FuzzyBASIC 料理法〈2〉
■86年12月号
第34部　CASL & COMET
連載　FuzzyBASIC 料理法〈3〉
■87年1月号
第35部　マシン語入力ツールMACINTO-C
連載　FuzzyBASIC 料理法〈4〉
■87年2月号
第36部　アドベンチャーゲーム MARMALADE
第37部　テキアペ作成ツール CONTEX
■87年3月号
第38部　魔法使いはアニメがお好き
第39部　アニメーションツール MAGE
付録　"SWORD" 再掲載と MAGIC の標準化
■87年4月号
第40部　INVADER GAME
第41部　TANGERINE
■87年5月号
第42部　S-OS"SWORD" 変身セット
第43部　MZ-700用 "SWORD" を QD 対応に
■87年6月号
インタラプト　コンパイラ物語
第44部　FuzzyBASIC コンパイラ
第45部　エディタアセンブラ ZEDA-3
■87年7月号
第46部　STORY MASTER
■87年8月号
第47部　パズルゲーム碁石拾い
第48部　漢字出力パッケージ JACKWRITE
特別付録　FM-7/77版 S-OS"SWORD"
■87年9月号
第49部　リロケータブル逆アセンブラ Inside-R
特別付録　PC-8001/8801 版 S-OS"SWORD"
■87年10月号
第50部　tiny CORE WARS
第51部　FuzzyBASIC コンパイラの拡張
第52部　X1turbo 版 S-OS"SWORD"
■87年11月号
序論　神話のなかのマイクロコンピュータ
付録　S-OS の仲間たち
第53部　もうひとつの FuzzyBASIC 入門
第54部　ファイルアロケータ&ローダ
インタラプト　S-OS こちら集中治療室
第55部　BACK GAMMON
■87年12月号
第56部　タートルグラフィックパッケージTURTLE
第57部　X1turbo 版 "SWORD" アフターケア　ラインプリントルーチン
特別付録　PASOPIA7 版 S-OS"SWORD"
■88年1月号
第58部　FuzzyBASIC コンパイラ・奥村版
付録　石上版コンパイラ拡張部の修正
■88年2月号
第59部　シューティングゲーム ELFES
■88年3月号
第60部　構造型コンパイラ言語 SLANG
■88年4月号
第61部　デバッギングツール TRADE
第62部　シミュレーションウォーゲーム WALRUS
■88年5月号
第63部　シューティングゲーム ELFES II
第64部　地底最大の作戦
■88年6月号
第65部　構造化言語 SLANG 入門(1)
第66部　Lisp-85 用 NAMPA シミュレーション
■88年7月号
第67部　マルチウィンドウドライバ MW-1
連載　構造化言語 SLANG 入門(2)
■88年8月号
第68部　マルチウィンドウエディタ WINER
■88年9月号
第69部　超小型エディタ TED-750
第70部　アフターケア WINER の拡張
■88年10月号
第71部　SLANG 用ファイル入出力ライブラリ
第72部　シューティングゲーム MANKAI
■88年11月号
第73部　シューティングゲーム ELFES IV
■88年12月号
第74部　ソースジェネレータ SOURCERY
■89年1月号
第75部　パズルゲーム LAST ONE
第76部　ブロックゲーム FLICK
■89年2月号
第77部　高速エディタアセンブラ REDA
特別付録　X1版 S-OS"SWORD"〈再掲載〉
■89年3月号
第78部　Z80用浮動小数点演算パッケージSOROBAN
■89年4月号
第79部　SLANG 用実数演算ライブラリ
■89年5月号
第80部　ソースジェネレータ RING
■89年6月号
第81部　超小型コンパイラTTC
■89年7月号
第82部　TTC用パズルゲーム　TICBAN
■89年8月号
第83部　CP/M用ファイルコンバータ
■89年9月号
第84部　生物進化シミュレーションBUGS
■89年10月号
第85部　小型インタプリタ言語TTI
■89年11月号
第86部　TTI用パズルゲーム　PUSH BON!
■89年12月号
第87部　SLANG用リダイレクションライブラリ　DIO. LIB
■90年1月号
第88部　SLANG用ゲームWORM KUN
特別付録　再掲載SLANGコンパイラ
■90年2月号
第89部　超小型コンパイラTTC++
■90年3月号
第90部　超多機能アセンブラOHM-Z80
■90年4月号
第91部　ファジィコンピュータシミュレーションI-MY
■90年5月号
第92部　インタプリタ言語STACK

＊以上のアプリケーションは，基本システムである S-OS "MACE" または S-OS"SWORD" がないと動作しませんのでご注意ください。

〈予告編〉

ハードウェア工作入門

Misawa Kazuhiko
三沢 和彦

ハードの知識や工作の心得があれば，コンピュータをより積極的に活用することが可能です。そこで始まるのが入門者を対象としたハードウェア工作の連載です。これまでハードは難しくてと敬遠していた方もこの講座で自信をつけてください。

ハードウェア実習への誘い

いまから2年前の実話です。X1ユーザーのM君はパソコンで音楽演奏をしたいと思いました。それまでX1内蔵のFM音源ボードと付属のソフトウェアで十分楽しんではいたのですが，1音1音ステップごとに入力していくのはやはり大変でした。M君は自分である程度キーボードが弾けたので，手弾き演奏の内容をそのままパソコンにデータとして取り込めたら便利だなあと思いました。

ちょうどそのころ，デジタル楽器の統一規格であるMIDIが普及し始めてきて，MIDIキーボードの価格も急激に下がってきていました。X1でもMIDIが使えたら便利なのにと思ったのですが，なにぶんインタフェイスがサポートされていませんでした。大部分の人は諦めるか，あるいはどうしてもMIDIキーボードをパソコンにつなごうというのなら，インタフェイスとソフトウェアがサポートされているPC-9801などに買い換えるところでしょう。

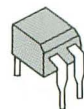
しかし，M君は諦めなかったのです。彼はMIDIキーボードを買ったあと，自らインタフェイスを設計，自作を試みたのです。もちろん最初はうまく動きませんでしたが，試行錯誤ののち，みごとにX1にMIDIキーボードをつないでしまったのです。

これはMIDI特集の前置きではありません。ここで皆さんにいいたいことは，我らがXユーザーには不可能を可能に変えるストロングタイプのマニアが多いということです。M君はハードウェアの知識があったおかげで，インタフェイスのないX1にMIDIをつなぐことができたのです。X68000などはそれ自体のアーキテクチャが非常に進んだものなので，プログラミングの面だけでも隅々まで使いこなすのはなかなか大変です。しかし，それ以上にハードの知識は，Xシリーズのポテンシャルを数倍にも引き上げます。

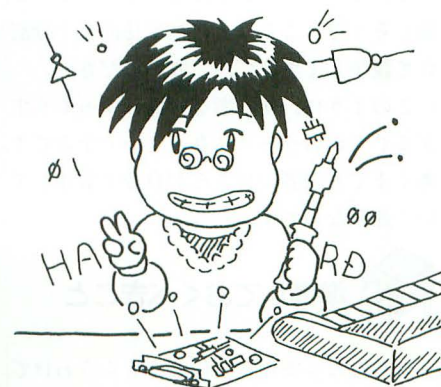
本誌でもお馴染みの栗野氏はハードにかけては数あるOh!Xスタッフのなかでも右に出る人はいません。栗野氏の製作記事を見ると，学習リモコンといい，ガイガーカウンタといい，比較的簡単な回路であれほどユニークなハードウェアを組み上げています。もちろん最初のアイデアがものをいうのですが，思いのままにボードを組み上げ，パソコンで制御するという楽しみ方に憧れを抱かないでしょうか？

いまはハードがさっぱりの皆さんでも第2，第3の栗野氏になれる可能性を秘めているのです。また，最近ではストロングな島田氏が，なんと自力でCPU（RISCプロセッサ？）まで組んでしまいました。実際あのRISCプロセッサには特殊な部品はなにひとつなく，身近に入手できるTTLの組み合わせだけでした。デジタル回路の基礎をかじった人なら，経験を少し積めばあの程度の回路を組むのも夢ではありません。そのほか，X1の8MHz化とか，PC-88VAの80386化とか，STUDIO Xでも一時話題になりましたが，こういった「禁断のパソコン本体改造」もハードウェアの知識次第なのです。

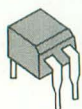
Oh!X誌上でも，これまでにハードウェアに関する記事もいくつか掲載してきました。私自身もデジタル回路入門の解説記事を手がけてきましたが，どうも解説を読むだけではよくわからない，という声を聞きます。

やはり，ハードをマスターするためには，読者の皆さんが自ら作業を行っていくのがいちばんです。そこで次号から，実際に簡単な回路を製作しながらハードの基礎から応用までをマスターしようという連載を始めることになりました。

この連載の方針

この連載で製作実習を行うハードは，X68000につなぐことを前提としています。単体で使えるものにしない理由はいくつかあります。まず，皆さんはパソコンユーザーなわけですから，ビデオやステレオに使われているハードではなく，パソコンにおけるハードウェアの考え方を学びたいところです。

しかし，コンピュータ内部をいきなりいじるのは，このあいだのRISCプロセッサ製作のように，初心者にはとてもついていけない内容になってしまいます。だからといって1つひとつの製作品を小さなものにすると，作っただけでまったく使えないものになりかねません。そこで，コンピュータにつなぐ外付けハードウェアのかたちにすると，実際に作る部分は小さくてすみ，それを実際に使うためにはパソコン側のプログラムで補ってやればよいわけです。

また，コンピュータのハードのなかでも，外部機器とデータをやり取りする「インタフェイス」について理解するだけでも有益であり，大切なことでもあります。さらに，プログラミングするうえでも，マシンをブラックボックスとしてとらえるのではなく，ハードの実際にどの部分が動作しているのか確認しながらアプリケーションを組むこともでき，マシンを文字どおり「操っている」感覚が実感できます。

ところで，X68000以外のユーザーを見捨てるのか，という意見も出てくると思いますが，この連載で製作するものは，パソコンの「ジョイスティックポート」につなぐので，X1にも少しの変更で利用可能です。それでは，なぜ「ジョイスティックポート」かというと，そこがいちばん簡単に接続できるインタフェイスだからです。

栗野氏の学習リモコンやガイガーカウンタもジョイスティックポート経由でつないでいます。初回でインタフェイスの基礎を解説するときにそのあたりの仕組みを詳しく解説します。実際に掲載する実体配線図やアプリケーションプログラムはX68000用ですが，X1用の変更点についてもその都度フォローするつもりです。

カリキュラム

では，連載講座で製作する予定のハードを紹介しましょう。

1) SW&LED基本I/O
2) A/Dコンバータ
3) センサ応用回路
4) リレー制御とロボット
5) 周波数カウンタ

入門編の内容としては以上です。

1)の基本I/Oはそれだけではあまり実用性があるとはいえませんが，パソコンのインタフェイスについてきっちり押さえておくことはとても重要なのと，実際に工作するのに慣れるのにもよいと思い，簡単な回路を選びました。ウォーミングアップのつもりでチャレンジしてください。

2)では，いきなり高度な内容になってしまうのですが，その分実用性も高いA/Dコンバータを扱います。自然界におけるデータはすべてアナログ量といってもよいので，これらのデータをパソコンに取り込むA/Dコンバータはパソコンインタフェイスのなかでももっとも重要なものです。普通は工作も大変複雑なのですが，部品を工夫して，誰にでも実現できる回路にしました。

3)のセンサは，2)で製作したA/Dコンバータに回路を外付けするかたちになっています。自然界のデータを実際に取り込む部分について詳しく研究します。

2)，3)ではデータの取り込みに中心のある製作でしたが，4)の制御では，パソコンから外部機器をコントロールする手法について実習します。現在世間で進められているオートメーション化のごく基礎を実現する回路を選びました。

5)は応用編に続く内容として，少し高度な回路を選びました。2)のA/Dコンバータと同じく測定器の基本ですから，応用例とあわせてじっくり製作していきたいと思います。

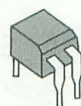
もしこんな回路を作ってみたいという要望があれば，連載中でも変更することがありますから，どんどん意見をお寄せください。また，応用編ということで連載を続けていってもよいと思います。

次に，連載の進め方について説明しておきましょう。ひとつの製作ハードにつき3部構成になっています。まず初回はこれから製作しようとするハードについての基礎知識について詳しく解説します。ただ作るだけというのではなく，自分が製作するものの仕組みをしっかり理解していくのが大切です。このときに具体的な回路図も挙げ，その回路の原理と特徴も説明します。

ほかにも，回路上で工夫の余地はないのか，なぜこの回路を選んだのかなどといった細かい点も説明できたらいいと思っています。

2回目は製作実習編で，部品集めから始まって実際に作業するうえでの手順と注意事項などを説明します。各部品の特徴と部品の選び方など，まったくの初心者でも部品屋に行って困らないことを目標にします。実体配線図なども可能な限り掲載していくつもりです。さらにここは配線のコツや工作のノウハウを伝授するところでもあります。これまでの製作記事では，回路図が載っていてもいざ作るとなるとどうしてよいかわからない，ということが多かったと思いますので，私も実際に製作しながら同時に解説記事にしていく感覚で説明するよう心掛けるつもりです。

最後はアプリケーションプログラムの掲載です。X68000上で一緒にプログラムを組みながら，製作した回路の実際の応用例を説明していきます。プログラムの記述はX-BASICで統一し，コンパイルしても動作するようにするつもりです。ただし，最初の基本I/Oの製作のところでI/Oドライバを外部関数のかたちで組み込みますが，これだけはアセンブラで記述します。しかし，難しそうなところはその都度誌面を十分割いて説明しますので，ご安心ください。

このように，初心者を対象にゆっくりすぎるぐらいのペースで進めていく予定です。あくまでも読者の皆さんが自分で製作していく連載なのです。

準備しておくべきこと

皆さんにも実際に工作してもらうわけですから，まず工具を揃えてください。工具

はケチると工作に苦労するうえ，できばえも悪いものにしかなりません。最低限必要なのは，ハンダゴテ，ニッパ，ラジオペンチ，カッター，ドライバ，ワイヤストリッパ，ピンセットの7点です。そして，必要な測定器としてテスターがあります。

ハンダゴテは，グットというメーカーのKS-20Rという型のものが手頃です。皆さんはこの程度で十分ですが，IC工作専用のコテ先の細いものがあれば，それだけ使い心地はよくなります。私は同じグットのCX-30を使っています。ハンダもコテ先にあわせて細いものを選びます。

また，スポンジのついたコテ台もセットで揃えましょう。このスポンジでコテの先を綺麗にしながらハンダ付けしていくのですが，ハンダ付けはコテ先が汚いと，どんなに高級なハンダゴテを使っていても絶対に失敗します。

余裕があれば，ハンダ吸い取り器も買っておくとたいへん重宝します。これは，ハンダ付けに失敗したときに余計なハンダを吸い取ってくれるもので，これがあれば，少々の失敗があっても安心です。私はグットのGS-20を使っています。

ニッパとラジオペンチとワイヤストリッパとを別々に揃えることを奨めます。一見ペンチだけで代用できそうですが，それぞれの役割があり，兼用にするのと使い分けるのとで作業の能率が全然違います。ニッパは導線を切断するためのもの，ワイヤストリッパは被覆付き導線の被覆をむくためのもの，またラジオペンチは針金などを曲げたり切ったりするためのものです。それぞれ1,000～1,500円程度のものでよいでしょう。参考までに私が使っているのは，ニッパはグットのYN-2，ラジオペンチはHOZANというメーカーのP-15，ワイヤストリッパはやはりグットのYS-2です。

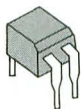
IC工作は部品が小さく作業も細かいので，ピンセットも必需品です。これは専用のものでなく，文房具屋で手に入るもので構いません。ただし，先の細いものでないと使えません。カッターも普通の事務用のもので十分です。

ドライバはプラスとマイナスの太さの違うものをそれぞれ3本ずつぐらい用意するとよいでしょう。連載では必要ありませんが，余裕があれば，手でにぎるタイプのもののほかに精密ドライバ（腕時計の修理などに使うもの）と呼ばれるセットも買い揃えておくと便利です。

次にテスターですが，これは趣味で電子工作するだけの人でも絶対にほしい測定器です。テスターは，電圧計と電流計と抵抗計とがスイッチひとつで切り替えられて，非常に便利です。デジタル回路の製作では，ロジックICの各端子の電圧を計り，HかLかを調べて，回路が正常に動作しているかをチェックするのに使われるのがほとんどです。

そのほかにも，配線がきちんと導通しているか，あるいは逆に余計なところがショートしていないかチェックするのにも必要です。値段も性能もさまざまなものが市販されていますが，初めてテスターを買う人には，ごく普通のアナログタイプ(指針式)で十分です。私は，SANWAというメーカーのものを使っています。

工具が揃ったら，次は部品集めです。秋葉原や日本橋が近い人でなければ，通信販売に頼るしかありません。連載では必ず部品表を載せますので，それをそのままコピーして販売店に送れば，まず間違いがないでしょう。品切れの場合でも店の人が代替品を選んでくれるように，回路図も添付すると万全です。

販売店はCQ出版社のトランジスタ技術という雑誌の広告欄を見てください。品揃え，スタッフの知識という点では，T・Z ONEパーツショップがもっともよい店だと思います(☎03-257-2655)。工具も取り扱っていますし，まずなんでも揃います。私自身も連載のために使うパーツはこの店で買えるものに限るつもりです。

参考文献

実際に工作を始める前に参考文献で勉強しておきたいという熱心な人もいるでしょう。まずは手前ミソですが，本誌1989年1月号の特集記事「ANDもORもこわくない」をすすめます。この記事でひと通りデジタル回路の基礎知識を得ることができるでしょう。ただ，誌面の制限もあって細かいところまできちんとフォローしているとはいえません。また，かなり以前のものなのでバックナンバーをお持ちでない方も多いでしょう。

そこで，内容的には少し高度ですが，きちんと書かれているものとして私もよく参考にしているのは，「ディジタルIC回路の設計」（湯山俊夫著，CQ出版社）です。これは，デジタル回路ではポピュラーなロジックICの動作と使い方とを丁寧に解説してあり，また利用価値の高い回路例も多くあります。この本は各ロジック回路を機能別に章立ててあり，この本をひと通りマスターすると，デジタル回路をIC 1個ずつ組み合わせてロジックを組める実力がつきます。

もう少し初心者向けとして，「絵ときディジタル回路」（内山明治，堀江俊明共著，オーム社）を挙げておきましょう。この本は，豊富な図解でデジタル回路の基礎知識をやさしく解説しています。

このほか，デジタル回路を設計製作する人には，TTL規格表（CQ出版社）が必携の書です。初心者にとってはこれだけ読んでもあまりよくわからないと思いますが，連載中でも説明に使いますので1冊揃えておくのがよいでしょう。TTLロジックICは型番が機能と1対1に対応しています。この規格表では型番順に並んでいて，それぞれのロジックICの足と入出力機能との対応が図表になっています。私も，この規格表がなければ手も足も出ないのです。

＊　＊　＊

さあ，来月号からは皆さんが主役です。いままで，ハードウェアなんて高嶺の花と思ってあきらめていた人でも，これからは気軽におつきあいください。連載が終わるころには，知らず知らずのうちにハードの基礎をマスターしていることに気づくことでしょう。

INTEGRAL X1

亀田 雅彦 Kameda Masahiko

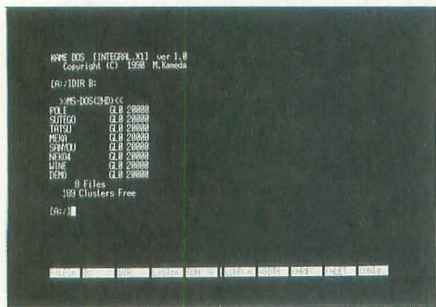
MS-DOSやHuman68kの環境をX1turboでも！ 今回からX1turbo用コマンドシェルシミュレータ作成講座の新連載です。メディアのコンバートだけでなくファイルの起動やさまざまな処理が可能。X1にも対応予定ですので，ご期待ください。

ハイ，皆さんこんにちは！

「X1の上にも6年」の異名をとる亀田です。いままで，ゲームレビューやったりX1のゲーム作ったりして，細々と機会をうかがっておりましたが（皆さんチェックしてくれてましたか？），そろそろ出番が回ってきたようです。X1の存亡をかけた，起死回生・一発逆転プログラム！ 怒濤の割り込み連載！ 構想に半年，制作に半年の歳月をかけて，いまここに，KAME-DOS，通称「INTEGRAL X1」を発表させていただきます。

世の中を見渡すと，いつのまにか32ビット＆ハードディスクが当たり前の時代になってしまいました。そんな中で，X1シリーズは今後どうすればよいのか？ そんなことを常々考えあぐねた末，ひとつの解答としてこの「INTEGRAL X1」が，そしてその奥にある「大同団結フォーマット」の発想が生まれました。

時代はパソコン乱立期から，徐々に標準化へと移行しています。パソコン自体の性能もさることながら，データ（情報）の重要性が増していますし，不特定多数のユーザーがパソコンを使うようになりました。そうなれば，（操作＆データの）統合化の道は避けて通れません。

逆にいえば，統合化した環境を持つ機種ならば，性能面で劣っていても使いものになるということです（注1）。

さて，そこでX1を考えてみますと，やっぱりシャープの独自色が強かったりします。CP/Mでは，それ自体もはや時代遅れです。そのような，X1のディスク状況の隙間を埋めるようなプログラムが，このKAME-DOSです。なお，今月は入門編でして，来月から連載させていただきます。

正体見たり！ INTEGRAL X1

はたして，謎のベールに包まれたKAME-DOSとはなんなのでしょう？ とりあえず，宣伝文句を列挙しておきます。

MS-DOSファイルも自由自在！
操作性はMS-DOS準拠！
BASICでアプリケーション・拡張性もバッチリ！ しかも高速！
X1シリーズフルサポート！（予定）
Oh!Xを買うだけでついてくる！

このKAME-DOSにいちばんイメージとして近いのは，以前本誌でも取り上げたことのあるマルチプラン付属のMS(X)-DOSでしょう。そのMS(X)-DOSは実はほとんどおまけという感じで，MS-DOSの2Dのみのサポートでした。

しかし，KAME-DOSでは，気合いを入れてMS-DOSとX1のそれぞれ2D・2HD計4種類をサポートしています。しかもそれらは，ほぼ区別することなく扱えるので，ユーザー側ではほとんど意識する必要がありません。ちょうどX68000のHumanがMS-DOSファイルを読み込めるように，（メディアが合えば）X1でもそれが可能になるのです（念のため：ファイルは読めるが，V30のプログラムが動くなんてことは絶対にない）。

もうひとつ見逃せない点は，KAME-DOSがBASIC＋マシン語で書かれているということです。通常，DOSのようなハードに密着したプログラムはコンパイラやアセンブラで組みます。しかし，X1の主力開発言語がBASICであることを考えると，こうするのがもっとも的確だと判断しました。おかげで，日本語がきわめて簡単に扱えますし，外部コマンドなども容易に開発できるのでした。ユーザーにとっても，簡単に自分用にカスタマイズしたDOSになります。これぞ，オープンアーキテクチャ！ と自己満足しております。

入力・実行方法

初期設定では，Zの2HDディスクに合わせてあります。もし，読み込めないことがありましたら，INTEGRAL.Xファイルの中の，1150行の‘&H10’をもう少し大きくしてみてください。また，ステップレートが3ms以外のドライブを使っている方は，1120行の注釈を取って1130行に注釈をつけてください。

2HDのディスクドライブをお持ちの方は今月号付録のディスクからプログラムディスクを作ることもできます。

オペレーション機能全開！

このDOSはかなりの変わり種ですが，エンドユーザーとして使うだけなら，難しいことはありません。そこで，ここから実際の機能とオペレーションを解説していきますが，その前にDOSについて触れておきましょう。

DOSとは「ディスクオペレーティングシステム」の意味で，フロッピーディスクをガシガシと読んだり・書いたり・フォーマットしたりするシステムプログラム（の集まり）です。

注1：PC-9801がいい例。性能で売れるならTOWNSやX68000が市場を独占しているはず。相変わらずのMS-DOSが，（アプリケーション）ユーザーにとっては実にありがたい。売れ筋に合わせるのが市場原理なので，KAME-DOSでもCOMMAND.COMをモデルにさせていただきました。なお，文中のMS-DOS（COMMAND.COM）は，Human（COMMAND.X）と読み換えてもらっても構いません。

ですから，CP/MもMS-DOSもS-OSもBASICも（ある意味では）みんなDOSです。ユーザーが，やりたいこと（ファイル名を見るとか，コピーするとか）をコマンドとして与え，それに従ってDOSが動作を実行していきます。

また，DOSを使うと面倒なディスク入出力をまかせられるし，ファイルの共通化も図れるという特典がつきます。16ビット以上の世界では，ファイルの共通化の概念は当たり前になりました(注2)。

その1 MS-DOS入門

KAME-DOSはMS-DOSそっくりです。で，説明終わり！　なら簡単なんですが，実際そうもいきません。そこで,「信じればわかる！　KAME-DOS入門」を始めます。CP/MやMS-DOSを知っている方は，ここを読み飛ばしても構いません。

DOSを起動すると,プロンプトとして［A：/］が君を出迎えてくれます。カーソルが点滅して入力を待っているようなので，とりあえず，

　　DIR

とでも打ってリターンキーを押すと，Aドライブのファイル名がプリントされましたね。これはBASICのFILESなのです！　Aドライブに運悪くディスクがささってないと，エラーを出します。あとは，こんな動作を無限連鎖地獄のように続けていくだけです。さあ，ここまでわかればもうだいじょうぶ。次に解説してあるコマンドをバシバシ打ち込んで実行してみよう。

その2 コマンド総集編

●予備知識

基本的に，プロンプトの隣から，

　　コマンド　ファイル名1　ファイル名2

の順で，それぞれスペースで区切ります。これらは，省略されることもあります。また，オプションはどこのあいだに入れても構いません。BASICと違って，ファイル名を囲うのに"(ダブルクォーテーション)は使いません。どの場合でも，大文字，小文字の区別はありません。ファイル名1が入力で，2が出力です。

例）　COPY　A：*.X1　B：

●ドライブ名

BASICの0：～3：がA：～D：，MEM0：MEM1：がE：F：，EMM0：～EMM3：がW：X：Y：Z：です。KAME-DOSでは，X1turboのバンクRAM をもドライブとしてサポートしているので，バンク0～バンク15がH：～V：になります。つまり，これ以外のドライブはサポートしていないというわけですね。

プロンプトに出てくるドライブ名は，現在自分のいるドライブです（カレントドライブ)。

●ファイル名

　［ドライブ：］［パス名］［ファイル名］

の順で，コマンドによっては省略可能なものもあります。ディレクトリの区切り記号は，BASICと同じく/(スラッシュ)です。なお，階層化ディレクトリの詳細については，マニュアルなり解説書なりを見てください(注3)。

もちろんワイルドカード（*と?）も使えます。'*'は，それ以降のファイル名はなんでもよいことを意味し，'?'はその位置の1文字がなんでもよいということです。

例）　A：/BIN/FORMAT. X1
　　　W：A*.*

/オプション/

　－英字1［数字1桁］［英字2……

頭にハイフン（マイナス記号）をつけたアルファベット1文字で，オプションを指定します。アルファベットの直後に数字（0～9）をつけることもあります。連続指定ができます。

例）　－T1A

さあ，やっと個々のコマンド解説です。

●X：

ドライブ名：(コロン)で，カレントドライブを変更します。

●DIR

　DIR　［ファイル名］［－W］

BASICでいうFILESです。ファイル名は省略可能です。写真を見てください。ファイル名(1)・ファイルの大きさ（16進数)

注2：MS-DOSのテキストファイルには，ワープロだろうがエディタだろうが互換性があったりします。PC-9801のエディタで書いた原稿が，X68000で修正できるのは実にありがたいのであります。プログラムを書くのにも，自分の好きなエディタが使えるので気持ちがよろしい(BASICのエディタはどこでもタコです)。

リスト1　オマケディスクからファイルを取り出す

```
1000 '
1010 ' MAKE [INTEGRAL X] from 68 DISK
1020 '
1030 '
1040 CLS:CLEAR &HE000:DEFINT a-z
1050 din =0                          '68 DISK IS 0:
1060 dout=1                          'X1 DISK IS 1:
1070 DEVICE STR$(dout)+":2":ro=48   'X1 DISK IS 2HD
1080 PRINT "MAKE [INTEGRAL X] PROGRAM":PRINT
1090 COLOR 4:PRINT "  X68000   DISK =";STR$(din );":"
1100 COLOR 5:PRINT "  X1turboZ DISK =";STR$(dout);":":PRINT
1110 COLOR 7:PRINT "   Ready ? (y/n)";
1120 i$=INKEY$(1):IF INSTR("Yyﾝ",i$)=0 THEN 1120
1130 PRINT:ri=11
1140 d=INP(&HFFE):OUT &HFFC,&H80+din:GOSUB "wnbsy"
1150 OUT &HFF8,0:GOSUB "wnbsy":OUT &HFF9,0
1160 FOR j=0 TO 19
1170 a=ri MOD 16:IF a>7 THEN sc=a-8:sd=&H10 ELSE sc=a:sd=0
1180 d=INP(&HFFE):OUT &HFFC,&H80+sd+din:GOSUB "wnbsy"
1190 OUT &HFFB,INT(ri/16):OUT &HFF8,&H1C:GOSUB "wnbsy"
1200 dma$=HEXCHR$("83 7D FB 0F FF 03 2C 10 80 8D 00 E0 92 CF 87")
1210 GOSUB "setdma":OUT &HFFA,sc+1:OUT &HFF8,&H80
1220 GOSUB "wnbsy" :GOSUB "reset"
1230 FOR i=0 TO 3
1240  DEVO$ STR$(dout)+":",ro,MEM$(&HE000+i*256,128),MEM$(&HE080+i*256,128)
1250 ro=ro+1:NEXT
1260 OUT &HFF9,INT(ri/16):ri=ri+1
1270 NEXT
1280 fat=28:dir=32
1290 DEVI$ STR$(dout)+":",fat,a$,b$:MEM$(&HE000,128)=a$:MEM$(&HE080,128)=b$
1300 MEM$(&HE000,16)=HEXCHR$("01 8F 8F 84 05 8C 07 8A 00 00 00 00 00 00 00 00")
1310 DEVO$ STR$(dout)+":",fat,MEM$(&HE000,128),MEM$(&HE080,128)
1320 DEVI$ STR$(dout)+":",dir,a$,b$:MEM$(&HE000,128)=a$:MEM$(&HE080,128)=b$
1330 MEM$(&HE000,16)=HEXCHR$("02 49 4E 54 45 47 52 41 4C 20 20 20 20 20 58 20")
1340 MEM$(&HE010,16)=HEXCHR$("20 20 28 04 00 00 00 00 90 55 18 12 00 00 03 00")
1350 MEM$(&HE020,16)=HEXCHR$("01 46 44 43 20 20 20 20 20 20 20 20 20 20 4F 42")
1360 MEM$(&HE030,16)=HEXCHR$("4A 20 00 1D 00 D0 00 D0 90 55 18 12 00 00 04 00")
1370 MEM$(&HE040,16)=HEXCHR$("02 43 4F 4D 4D 41 4E 44 20 20 20 20 20 20 58 31")
1380 MEM$(&HE050,16)=HEXCHR$("20 20 A4 1A 00 00 00 00 90 55 18 12 00 00 06 00")
1390 DEVO$ STR$(dout)+":",dir,MEM$(&HE000,128),MEM$(&HE080,128)
1400 DEVICE STR$(dout)+":":PRINT:FILES:PRINT
1410 CFLASH 1:PRINT "COMPLETED !!":CFLASH 0:END
1420 '
1430 LABEL "reset"
1440  dma$=HEXCHR$("83"):GOSUB "setdma":RETURN
1450 LABEL "wnbsy"
1460  ct=0
1470  IF ct>1000 THEN OUT &HFFC,din:PRINT "DISK?":STOP
1480  s=INP(&HFF8):IF s AND &H81 THEN ct=ct+1:GOTO 1470
1490 RETURN
1500 LABEL "setdma"
1510  FOR ii=1 TO LEN(dma$):OUT &H1F80,ASC(MID$(dma$,ii,1)):NEXT:RETURN
```

(2)・属性(3)・[スタートアドレス](4)の順でディレクトリが表示されていますね。属性は,

P……BASICのプロテクトファイル
D……下位ディレクトリ
M……マシン語ファイル
B……BASICファイル
A……ASCIIセーブされたファイル

です。MS-DOSファイルにはP以外の属性はありません。スタートアドレスが表示されるのは,マシン語ファイルのみです。オプションにはW(ワイド)がありまして,ファイル名のみずらずらと続けて出力します。

例) DIR, DIR -W, DIR A:, DIR*.X1

●TYPE

TYPE [ファイル名] [-AT]

ファイルの中身を表示します。BASICのASCII形式はもとより,S-OSやMS-DOSファイルも自動的に判断して,うまい具合に表示するので心配いりません(注4)。表示している途中,[スペース]で一時停止し,一時停止中[ESC]で表示を終了します。

Aオプションは,BASICでいうとPRINT#0で表示せよ! ということです(普通はただのPRINT)。この違いは試してみれば一目瞭然でしょう。

Tオプションについては,Tオプションの説明を見てください。

例) TYPE AUTOEXEC. BAT, TYPE A:*.DOC

●DEL

DEL ファイル名

BASICのKILLです。

なおHuman68kのように実行に先立って,いちいち確認を求めてこないので注意してください。

例) DEL TEST. BAS, DEL A:*.*

●CD

CD パス名

CHDIR(チェンジディレクトリ)のことです。基本的な指定方法はBASICと同じですが,“..”がひとつ上のディレクトリを指しています。

下位ディレクトリにいるときにディスク交換をすると,表示が狂うことがあります。そのときは一度ルートへ戻ってください。

例) CD SYS, CD/BIN/BASIC/, CD .., CD /

●COPY

COPY ファイル名1 [ファイル名2] [-T]

ファイルのコピーを行います。BASICと同じ順序で,1から2へのコピーとなります。第2ファイル名を省略すると,カレントドライブへのコピーです。コピー先に同一ファイル名がある場合は,確認なしにもともとあったファイルを消去するので注意してください。MS-DOSではファイルの連結もできましたが,このDOSではできません。

Tオプションは,TYPEのそれと同じ働きをします。

例) COPY A:*.X1 E:, COPY *.BAS *.DOC

以上,内部コマンド(組み込まれているので,いつでもどこでも実行できる)でした。これ以外はすべて外部コマンドになります。外部コマンド(トランジェントコマンド)とは,コマンドをファイルとして保存しておいて実行時に読み込む方式のことです。今月はまだDOS専用の外部コマンドはありませんが,通常のBASICプログラムをDOSから実行できるようになっています。これについては,後ろの基本機能講座を見てください。

デバイスドライバとリダイレクション

これらは,MS-DOSなどで採用されている機構です。もし,あなたがファイルの中身をプリンタに出力したいと思ったらどうしますか? そういうときは,

COPY README. DOC PRN
TYPE README. DOC >PRN

などとしましょう。また,DIRの出力を画面ではなくプリンタにするには,

DIR >PRN, DIR A:*.X1 >PRN

というふうに指定します。つまり,ファイル名としてNUL, CON, AUX, PRNを与えると,そのデバイスに対して入出力するのです。

また,DIRのように出力ファイル名を指定しないコマンドでは,頭に“>”(不等号)をつけます(リダイレクション)。

NUL……(入力)なにも入力しない
　　　　(出力)なにも出力しない
CON……(入力)KAME-DOSではサポートなし
　　　　(出力)画面へ出力
AUX……入出力ともにサポートなし
PRN……(入力)できない
　　　　(出力)プリンタへ出力

例) DIR A:*.X1 >NUL
COPY README. DOC CON

つまり,上記の3文字がファイル名の場合はアクセスできません。

なお,DIRをプリントアウトしても,ファイルの個数・ディスクの残量などは画面に出力されます。また,プリンタ関係でエラーが出ると,COMMAND. X1は止まってしまうのでもう一度RUNしてください(しっかり手抜きしてます)。

MS-DOSのリダイレクションではファイルを作成することもできましたが,KAME-DOSではできません。

また,入力リダイレクション(<)もサポートしていません。

注3:ファイル名は,それぞれのシステムと同じです(MS-DOSは8+3文字)。漢字を入力する方法は,普通のBASICのやり方(SHIFT+XFER)です。X68000は小文字のファイル名も許しますが,MS-DOSではアクセスできないので注意してください。また,‘>’もファイル名には使えません。

注4:うまい具合というのは,改行コードとかコントロールコードとかを変換してくれるという意味です。同じフォーマット同士ではなにもしませんが,異なるフォーマット同士では,0DH+0AHと0DH(改行コード)変換,&H20以下のコードを&H20にする変換を同時に行います。S-OSはENDコードが違うだけなので変換はしません。

Tオプションの説明

Tオプションは,データ変換の際どのような変換を行うかを指定します。Tの後ろにつける数字によって変換方法が決まります。ちなみに,なにも指定しないと1を指定したことになります。下記のEOFとは,CTRL+Z(&H1A)のことです。

T1:ファイルサイズやEOF, その他のコードをすべて無視して,FATの分だけきっちり転送します。
T2:EOFのみチェックしながら,FAT分の転送。
T3:0DH+0AH→0DH, &H20以下,EOFをチェックしながらFAT分の転送。
T4:0DH→0DH+0AH, &H20以下,EOFをチェックしながらFAT分の転送。

T5~T8は,T1~T4をFAT転送ではなく,ファイルサイズ転送にしたもの。ディスク上でのデータの長さより(FAT),ディレクトリに書かれているファイルサイズが小さい場合,ファイルサイズを優先するということ。

オプションを指定しないと,同一フォーマット間転送ではT5が,異機種フォーマット間ではT7かT8が選択されます。また,FATとファイルサイズがあまりにも違いすぎる場合は,FATを優先します。

こんな面倒くさい指定があるのは,BASICがファイルサイズに2バイトしか取っていないからです。だから,ASCファイルの大きさはみんな0だし,辞書の大きさもわけのわからん数字だったりするのです。

X1→MS-DOS, MS-DOS→X1のCOPYをするときは,どんなファイルをコピーするのかよく考えましょう。

パワフル機能満載！ 基本機能講座

KAME-DOSの基本はMS-DOSもどきなので，BASICやCP/Mには見られない機能があったりします。それをここで，まとめてどどーんと公開していきましょう。MS-DOSの解説になりがちなのは，見逃してください。

●ディスク自動判別機能

これは今月の超目玉商品です！ Zなど，2D/2HD両用タイプのディスクドライブを持っている方（Zなどの内蔵ドライブ）は，BASICのDEVICE命令にハマッたことがあるでしょう。いちいちユーザーが2D・2HDの指定をしてやらなきゃならないし，挙句の果てに，間違えるとえんえんとディスクを回して，やっとエラーが出て止まるほどです。PC-9801のMS-DOSでは自動的に見分けてくれるのに，どうしてZでできないのかっ！ という怒髪天パワーを胸に，X1・XS-DOSの2D・2HD を自動的に判別する機能をつけました。

ディスクアクセスするときは必ずディスクをチェックして，そのフォーマットに合わせますから，ユーザーはただ目的のディスクを挿入すればいいだけです。ちなみに，いくら自動判別といっても，2D専用ドライブで2HDが読めないのはいうまでもありません。

●BASICファイル実行機能

せっかくBASIC上で動いているKAME-DOSですから，ほかのBASICプログラムをDOSからRUN させたいもんです。コマンドライン（プロンプトの出ている状態）から，BASICプログラムのファイル名を打ち込むと，そのプログラムをRUNします。拡張子はいらないので，FORMAT.BASならFORMATと打ち込んでください。

この機能で，ファイル名がそのまま外部コマンドの名前になっちゃうのでした。なお，このDOSの外部コマンドの拡張子は～.X1 です。それ以外の普通のファイルも実行できますが，DOSに帰ってくることはできません。

MS-DOSでは拡張子による実行の優先順位がありましたが，KAME-DOSでは PATHに従って最初に見つかったファイルを実行します。

それから，システムRUN 以外のマシン語ファイル・H：～V：(バンクメモリ）にあるファイル・MS-DOSディスクは実行できません。マシン語領域に&HD000からを取っているので，フリーエリアの関係で動かないプログラムもあります。

●テンプレート機能

MS-DOSでは，直前の入力内容をもう1度打たなくてもいいように，テンプレートなるものがあります。それとちょっと違いますが，直前に入力した内容を，ファンクションキーに定義してしまおうという安易な発想です。F3にコマンド部分，F2にファイル名1の部分が定義されるので，利用してください。

●コンバート機能

以前，Oh!XがOh!MZだった頃，「データの互換性を探る」と題して，MS-DOSとBASIC のファイルコンバータが発表されていました。その思想の流れを汲むのがこのコンバート機能です。この2つのフォーマット間のコンバートは，自動判別機能のおかげで，違いをそれほど気にする必要はありません。問題は，データ形式の違いです。コントロールコードと改行コードの処理の部分はなかなか面倒くさいのですが，Tオプションの説明を見て理解してください。

ASCII形式のファイル（TYPEしてちゃんと読めるファイル）はそのままCOPYしてください。グラフィックなどの生データ（加工したくないデータ）は，基本的にオプションにT0を指定してCOPYすればよいでしょう。

なお，ファイルの最終更新年月日のコンバートは行いません。

●PATH機能

ファイル名を入力して，そのファイルを実行させるときに，カレントドライブにそのファイルがないと実行できないのでは不便です。そこで，「A：の次はB：，その次はC：を探せ」ということを前もって指定しておくと，カレントドライブ，A：，B：，C：の順でそのファイルがあるかどうか調べて，見つけたドライブからそのプログラムを起動するのがPATHの機能です。BASICでDEVICE“：1”などとするのが，DOSでいうカレントドライブの変更で，そこからまたほかに探しにいくのが，BASICにはないPATH指定なのです。

INTEGRAL.Xというプログラム中（MS-DOSのCONFIG.SYSに相当）で，PATHを指定します。プログラムを見るとわかりますが，

“ドライブ名：/[パス名/]；…
…；”＋CHR$(0)

という文字列を，メモリの&HEC80～に格納することで指定できます。先頭から探して，；(セミコロン）が区切り記号，CHR$(0) がエンドコードです。

例） P$＝“A：/SYS/；B：/；”：MEM$(&HEC80，LEN（P$))＝P$

仕事にも十分使えるみたい！ 実際編

なにごとでも，形から入るのが上達の近道です。ここでは，KAME-DOSを使ううえで普通に使われるワザを中心に説明します。なお，ここに書かれている以上のハイテクは，ユーザー自身の手で見つけていってください。

●ファイル名中のスペース飛ばし

“BASIC CZ8FB01.SYS” というファイル名を指定したいとします。KAME-DOSでは，スペースがあると即ファイル名の区切りと判断するので，そのままでは指定できません。こういう場合は，

BASIC＊またはBASIC？CZ8FB01 というふうに，ワイルドカードを使ってください。これは，いちいちファイル名を全部打ち込みたくない場合にも使えます。パス名には使えません。

●バッチ処理

コマンドラインからの入力には INPUT文を使っているので，KEY 0 命令を使ってバッチ処理もどきができます。INTEGRAL.Xファイルの中で，

KEY 0，“COPY ＊.X1 E：”
＋CHR$(13)

と，いうふうに定義しておけば，起動時に自動的にファイルを転送できます。

●BASICとの関連機能

DOSでは，カレントドライブを変更するとき，ついでにDEVICE命令を実行して BASIC のデフォルトデバイスも変更しています。つまり，ブレイクしたりほかの BASICプログラムを実行した際にも，DOSと同じようにデフォルトデバイスや2D・2HDが設定されているのです。これを利用すると，辞書をRAMディスクにコピーしたあとに，日本語入力モードで辞書設定ができたりします。

また，INTEGRAL.Xでは，高解像度（WIDTH80，25，0，2）で起動するよう設定されていますが，低解像度（WIDTH80，25，0，1）にしても動きます。ただし漢字は使えません。

それに，TYPEなどをよく使う場合は，turbo “SWORD”（1987年10月号）と一緒に発表された，DMA スクロールルーチンを組み込むことをおすすめします。普通のBASICのスクロールでは遅すぎて使いものになりません（そのために，&HED00から256バイト空けてあるんです)。

●バッファ関連

ディスクアクセスをするプログラムは，必ずメモリにバッファ領域を取らなければなりません。KAME-DOSでは，FATバッファに4Kバイト，データバッファに4Kバイト×2の合計12Kバイトが最低でも必要です。バッファを取るメモリは，メインメモリ・G-RAM・バンクメモリの中から選択できます。

INTEGRAL.Xの1210行で，&HD000がFATバッファ先頭アドレス。&HE000がデータバッファの先頭アドレス。1220行で，&H1000がデータバッファひとつの大きさです。S_IOMMの2は，MEM1：を指しています。詳しくは，図1を見て理解してください。なお，バッファとRAMディスクの領域は，重ならないように注意してください（この設定なら，F：の残り容量が3クラスタ以下にならないようにしてください）。

X1ユーザー目覚めよ!

BASICでDOSを作った！ なんていうと，結構キワものの印象があるかもしれません。でも，スピードその他の点に関して，「使える」ということを最重要課題にしたので，MS-DOSと比べても決して見劣りのしない操作性になったと思います。

もともとBASIC自体にDOSとしての機能があるんだから，相当なものでないと作るだけ無駄になってしまいます。当初は簡単なコンバータの予定だったんですが，DOSにまで発展したのもそんな思いがあったからです。だいたいCOMMAND.COM(MS-DOSのやつね）ぐらいだったら，X1でだって簡単だと思ったんですよ。さすがにVS（X68000ね）とまではいいませんけど……。

つい3年くらい前までは，いろんなパソコンが発表されて，いろんなDOSが生み出され，それに従って，さまざまなデータ形式，ユーザーインタフェイスが世の中に氾濫しました。この混迷の時代では，ユーザーは機種ごとに違う操作性を強いられたし，同じ5インチでありながらまったくアクセスできないなんてことが当然でした。でも，こんな間違った状態が長く続くわけありません。混乱はやがて収束して，いまではだいぶ一本化されつつあるようです（まだムカつくことはあるけど）。

やっぱり，そうなるのが時流であり，そうなるよう努力するのがユーザーやメーカーの務めだと思います。もちろん，機種ごとの個性まで否定してしまうわけじゃありません。そういうものを，もっと包括的にとらえてすべてを包み込むようなやさしいOSが，私の理想なんですが……。

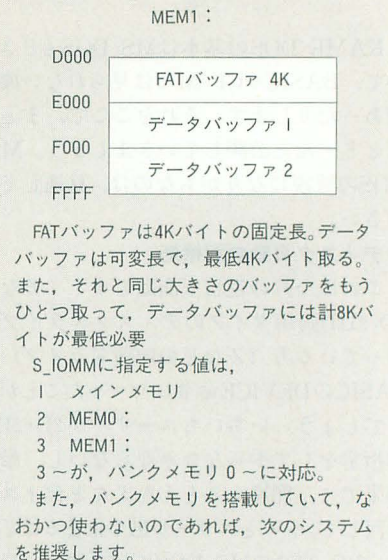
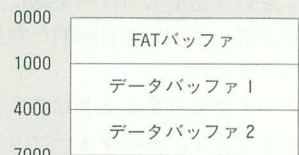
そういうわけで，ユーザーがこのプログラムによって，X1の存在意義を見つけてくれることを願っています。もし，このDOS上で「こんなことをやってほしい」とか，「こんなふうにしたほうがいいんじゃない？」とかありましたら，どしどし意見をお送りください。読者の方々と一緒に，プログラムを発展させていこうと思っています（要するに，まだ完成してないってこと）。

ではまた来月。

図1　バッファ概念図

FATバッファは4Kバイトの固定長。データバッファは可変長で，最低4Kバイト取る。また，それと同じ大きさのバッファをもうひとつ取って，データバッファには計8Kバイトが最低必要

S_IOMMに指定する値は，
1　メインメモリ
2　MEM0：
3　MEM1：
3～が，バンクメモリ0～に対応。

また，バンクメモリを搭載していて，なおかつ使わないのであれば，次のシステムを推奨します。

設定する値は，
MEM$(S_FF, 2)＝MKI$(&H0)
MEM$(S_BUFF, 2)
＝MKI$(&H1000)
MEM$(S_BSIZ, 2)
＝MKI$(&H3000)
POKE S_IOMM, 3
INTEGRAL. Xの1210行からを変更してください。

リスト2

```
1000 '
1010 'INTEGRAL.X1       CONFIG FILE
1020 '                                  (C) M.Kamdeda 1990
1030 '
1040 WIDTH 80,25,0,2:KLIST 1:KMODE 1:OPTIONSCREEN 4
1050 DEFINT a-z:CLS 4:INIT:SCREEN
1060 PRINT "KAME DOS  [INTEGRAL.X1]  ver 1.0"
1070 PRINT "   Copyright (C)  1990  M.Kameda"
1080 CLEAR &HD000:LOADM "FDC.OBJ"
1090 'm_rwrec=&HE000:MEM$(m_rwrec)=MKI$(&HF000)
1100 'm_wttrc=&HE01E:MEM$(m_wttrc)=MKI$(&HF003)
1110 v_rscmd=&HE0A6:v_skcmd1=&HE0A7:v_skcmd2=&HE0A8
1120 'POKE v_rscmd,2:poke v_skcmd1,&H1A:poke v_skcmd2,&H1E
1130  POKE v_rscmd,0:POKE v_skcmd1,&H18:POKE v_skcmd2,&H1C
1140 v_smacs=&HE0AB:POKE v_smacs,1
1150 v_wait=&HE0AC :POKE v_wait,&H10
1160 v_ctrl=&HD18E:s_path=&HEC80:s_s=&HECC0
1170 s_mac4=s_s   :s_tr4=s_s+26 :s_dn=s_s+30  :s_buff=s_s+31:s_ff=s_s+33
1180 s_bsiz=s_s+35:s_escp=s_s+37:s_iomm=s_s+38:s_eof=s_s+39 :s_eof3=s_s+40
1190 FOR i=s_mac4 TO s_mac4+26:POKE i,3:NEXT:POKE s_tr4,0,0,0,0
1200 INIT "MEM0:":INIT "MEM1:"
1210 MEM$(s_ff  ,2)=MKI$(&HD000):MEM$(s_buff,2)=MKI$(&HE000)
1220 MEM$(s_bsiz,2)=MKI$(&H1000):POKE s_iomm,2
1230 POKE s_escp,&H12:POKE s_eof,26:POKE s_eof3,26:POKE v_ctrl,32
1240 p$="e:/;f:/;w:/;a:/;b:/;"+CHR$(0):MEM$(s_path,LEN(p$))=p$
1250 POKE s_dn,0:POKE &HD07F,1
1260 RUN "COMMAND.X1"
```

リスト3

```
1000 '
1010 'INTEGRAL.X1  COMMAND FILE
1020 '
1030 '
1040 SCREEN:CLEAR:DEFINT A-Z:DIM mac(26),dir$(26)
1050 GOSUB 2610:'SAVE "MEM0:COMMAND.X1"
1060 DEFUSR0=m_opens:DEFUSR1=m_preop:DEFUSR2=m_setdn:DEFUSR3=m_tran
1070 DEFUSR4=m_alta
1080 FOR i=0 TO 26:mac(i)=PEEK(s_mac4+i):POKE v_csdir+i,0:NEXT:POKE v_dpnf,1
1090 dn=PEEK(s_dn):buff=CVI(MEM$(s_buff,2)):fb=CVI(MEM$(s_ff,2))
1100 POKE v_dn,dn:POKE v_mac,mac(dn):POKE v_wfd0,PEEK(&HF8D6)
1110 POKE v_iomm,PEEK(s_iomm):MEM$(v_badr,2)=MKI$(buff)
1120 bsiz=CVI(MEM$(s_bsiz,2)):MEM$(v_bsiz,2)=MKI$(bsiz):MEM$(v_ff,2)=MKI$(fb)
1130 '_______ MAIN
1140 '
1150 WHILE 1
1160  GOSUB 1230:MEM$(v_ff,2)=MKI$(fb):IF PEEK(v_stop) GOSUB 1190
1170  POKE v_dn,dn:POKE s_dn,dn:POKE v_mac,mac(dn):es=0:GOSUB 1520
1180 WEND:END
1190 '
1200 RESTORE 2910:FOR i=1 TO PEEK(v_stop):READ a$:NEXT
1210 BEEP:PRINT:PRINT a$:POKE v_stop,0:RETURN
1220 POKE v_stop,10:RETURN
1230 '
1240 IF tp2$<>"" THEN KEY 2,tp2$
1250 IF tp1$<>"" THEN KEY 3,tp1$+" "
1260 FOR i=0 TO 25:POKE v_alpa+i,0:NEXT:IF PEEK(&HD07F) THEN KLIST 1
1270 FOR i=0 TO 10:fe$(i)="":NEXT:POKE v_ddrv+1,7,1
1280 GOSUB 2490:GOSUB 2460:IF PEEK(v_stop) RETURN
1290 cm=0:a0=1:IF fe$="" THEN dn=PEEK(v_dn):RETURN
1300 REPEAT
1310  WHILE MID$(fe$,a0,1)=" ":a0=a0+1:WEND
1320  a=INSTR(a0,fe$," "):IF a=0 THEN a=LEN(fe$)+1
1330  d$=MID$(fe$,a0,a-a0):a0=a+1
1340  IF LEFT$(d$,1)="-" THEN op$=d$:d$=USR4(d$) ELSE fe$(cm)=d$:cm=cm+1
1350 UNTIL a>LEN(fe$):tp1$=fe$(0):tp2$=fe$(1)
1360 RESTORE 3040:i=0:REPEAT:READ a$:i=i+1:UNTIL a$=fe$(0) OR a$="end"
1370 ON i GOTO 1600,1600,1770,1770,1760,1760,1670,1670,2000,2000
1380 '
1390 i$=dir$(dn)
1400 fe$=fe$(0):od=0:op=2:sb=1:GOSUB 2310:IF PEEK(v_stop)=0 THEN 1420
1410 i$="":POKE v_od,0:POKE v_op,2:d$=USR3(fe$):IF PEEK(v_stop) RETURN
1420 k=PEEK(v_dn):dn=k:mac(k)=PEEK(v_mac)
1430 IF PEEK(v_p256) THEN i$=MEM$(v_p256+1,PEEK(v_p256))
1440 GOSUB 1520:IF i THEN POKE v_stop,11:RETURN
1450 IF PEEK(&HD07F) THEN d$=d$+y$
1460 fe$=MEM$(v_extn,3):m$=d$+i$+MEM$(v_fnam,13)+"."+fe$
1470 proces$(proces)="COMMAND.X1":proces=proces+1
1480 IF fe$="X1 " OR fe$="BAT" THEN 1490 ELSE 1500
1490 CHAIN m$
1500 KEY0,CHR$(26)+"RUN"+CHR$(34)+m$+CHR$(13):END
1510 '
1520 i=0:IF (5<dn AND dn<22) OR mac(dn)=2 OR mac(dn)=4 THEN i=1:RETURN
1530 d$=STR$(dn)+":":IF dn<4 THEN IF mac(dn)=1 THEN s$="2" ELSE s$="0"
1540 IF dn=4 THEN d$="MEM0:":GOTO 1580
1550 IF dn=5 THEN d$="MEM1:":GOTO 1580
1560 IF dn>21 THEN d$="EMM"+RIGHT$(STR$(dn-22),1)+":":GOTO 1580
1570 DEVICE d$+s$
1580 DEVICE d$:RETURN
1590 '_______ COMMAND
1600 '
1610 sb=0:op=1:GOSUB 2280:k=PEEK(v_stop)
1620 IF (k<>0 AND k<>3) OR PEEK(v_ddrv+1)<>7 RETURN
1630 PRINT:PRINT "  >>"+dms$(PEEK(v_mac))+"<<"
1640 CALL m_dirsb:ON PEEK(v_iofg) GOSUB 2120:IF es THEN PRINT
1650 PRINT "       ";PEEK(v_yen);"Files"
1660 GOSUB 2420:PRINT "  ";CVI(MEM$(v_fcrs,2));"Clusters Free":RETURN
1670 '
1680 sb=1:od=1:op=0:GOSUB 2310:k=PEEK(v_stop)
1690 IF k=3 THEN POKE v_stop,0:RETURN
1700 IF k OR PEEK(v_ddrv+1)<>7 RETURN
1710 WHILE PEEK(v_stop)=0
1720 IF PEEK(v_fdop+2)<>0 THEN PRINT MEM$(v_fnam,16);" Not DEL":GOTO 1740
1730 CALL m_dlfat:CALL m_dldir:PRINT MEM$(v_fnam,16)
1740 CALL m_dir2:WEND:POKE v_stop,0:CALL m_clos2:RETURN
1750 '
1760 POKE v_ddrv+1,7,7
1770 '
1780 k=0:t=0:POKE v_iofg,0:s=&H10
1790 POKE v_dn,dn:od=1:GOSUB 2330:m1=PEEK(v_mac):IF PEEK(v_stop) RETURN
1800 POKE v_dn,dn:od=2:GOSUB 2330:m2=PEEK(v_mac):IF PEEK(v_stop) RETURN
1810 i0=PEEK(v_ddrv+1):i1=PEEK(v_ddrv+2)
1820 IF (m1=2 OR m1=4) AND (m2=1 OR m2=3 OR i1<>7) THEN t=4:s=&H12
1830 IF (m1=1 OR m1=3 OR i0<>7) AND (m2=2 OR m2=4) THEN t=&H20:s=&H13
1840 i=PEEK(v_alpa+19):IF i THEN s=opt(i)
1850 IF t=4 AND s<>&H12 THEN t=1
1860 '
1870 sb=1:od=1:op=0:GOSUB 2390:IF PEEK(v_stop)=3 THEN 1990
1880 f$=MEM$(v_fbyt,4):IF PEEK(v_stop) RETURN
1890 sb=1:od=2:op=3:GOSUB 2390:IF PEEK(v_stop) RETURN
1900 POKE s_escp,s:k=k+1:IF PEEK(v_ddrv+1)=7 THEN PRINT MEM$(v_fnam,16)
1910 '
1920 CALL m_devi:IF PEEK(v_stop) RETURN
1930 ON PEEK(v_od) GOSUB 2110,2120:IF es OR PEEK(v_stop) RETURN
```

```
1940 IF PEEK(v_iofg) GOTO 1910
1950 '
1960 IF PEEK(s_escp)=0 AND t<>&H20 THEN MEM$(v_fszl,4)=f$
1970 POKE v_zoku+2,PEEK(v_zoku+1):IF t THEN POKE v_zoku+2,t
1980 GOSUB 2420:IF PEEK(v_stop) OR PEEK(v_ddrv+1)<>7 RETURN ELSE 1870
1990 POKE v_stop,0:PRINT k;"Files":RETURN
2000 '
2010 od=1:fe$=fe$(od):GOSUB 2460:k=PEEK(v_dn):IF fe$="" RETURN
2020 IF fe$=y$ THEN POKE v_csdir+k,0:dir$(k)="":RETURN
2030 d=0:IF PEEK(v_csdir+k)>0 GOSUB 2550
2040 IF d THEN POKE v_csdir+k,PEEK(v_csdir+k)-1:RETURN
2050 POKE v_cdf,1:GOSUB 2330:POKE v_cdf,0:d=PEEK(v_stop)
2060 IF (d<>0 AND d<>8) OR PEEK(v_ddrv+1)<>7 RETURN
2070 d$=MEM$(v_p256+1,PEEK(v_p256)):IF LEFT$(fe$,1)=y$ THEN dir$(k)=""
2080 dir$(k)=dir$(k)+d$:RETURN
2090 '_______ DRIVER
2100 '
2110 GOTO 1220
2120 '
2130 ON PEEK(v_ddrv+2) GOTO 2140,1220,2220:RETURN
2140 '
2150 w=CVI(MEM$(v_msbt,2)):i0=w ¥ 256:i1=w MOD 256:w=PEEK(v_alpa)
2160 FOR i=1 TO i0:CALL m_trs:FOR j=0 TO 3
2170  d$=MEM$(v_p256+j*64,64):IF w THEN PRINT#0# d$; ELSE PRINT d$;
2180  GOSUB 2520:IF es THEN i=i0:j=3
2190 NEXT:NEXT:IF es RETURN ELSE CALL m_trs:d$=MEM$(v_p256,i1)
2200 IF w THEN PRINT#0# d$; ELSE PRINT d$;
2210 RETURN
2220 '
2230 CALL m_gyoki:w=PEEK(v_p256):i0=PEEK(v_p256+1)
2240 GOSUB 2520:IF es THEN LPRINT:RETURN
2250 d$=MEM$(v_p256+2,i0):IF w=0 OR w=1 THEN LPRINT d$; ELSE LPRINT d$
2260 IF w=0 RETURN ELSE 2230
2270 '_______ SUB
2280 '
2290 IF LEFT$(fe$(1),1)=">" THEN fe$(2)=fe$(1):fe$(1)=""
2300 od=1:GOSUB 2310:IF PEEK(v_stop) RETURN ELSE od=2:GOTO 2310
2310 '
2320 GOSUB 2330:IF PEEK(v_stop) RETURN ELSE GOTO 2390
2330 '
2340 IF od=2 AND LEFT$(fe$(2),1)=">" THEN fe$(2)=RIGHT$(fe$(2),LEN(fe$(2))-1)
2350 POKE v_od,od:d$=USR1(fe$(od)):i0=PEEK(v_dn)
2360 IF PEEK(v_csdir+i0)=0 THEN dir$(i0)=""
2370 fe$(od)=RIGHT$(fe$(od),PEEK(v_yen)):IF PEEK(v_stop) RETURN
2380 IF PEEK(v_ddrv+od)=7 THEN mac(i0)=PEEK(v_mac):RETURN ELSE RETURN
2390 '
2400 POKE v_od,od:ON PEEK(v_ddrv+od)-6 GOTO 2410:RETURN
2410 POKE v_sbdr,sb:POKE v_op,op:d$=USR0(fe$(od)):RETURN
2420 '
2430 POKE v_od,2:ON PEEK(v_ddrv+2)-2 GOTO 2440,1220,1220,1220,2450:RETURN
2440 LPRINT:RETURN
2450 CALL m_saved:RETURN
2460 '
2470 d$=USR2(fe$):IF MID$(fe$,2,1)=":" THEN fe$=RIGHT$(fe$,LEN(fe$)-2)
2480 RETURN
2490 '
2500 fe$="":PRINT:COLOR 5:PRINT "[";CHR$(65+dn);":";y$;dir$(dn);"]";
2510 COLOR 7:INPUT "",fe$:RETURN
2520 '
2530 es=0:IF INKEY$<>" " RETURN ELSE IF INKEY$(1)=CHR$(27) THEN es=1
2540 RETURN
2550 '
2560 IF LEN(fe$)<2 RETURN ELSE IF LEFT$(fe$,2)<>".." RETURN
2570 IF LEN(fe$)=2 THEN d=1
2580 h=LEN(dir$(k)):a=1:REPEAT:b=a:a=INSTR(b+1,dir$(k),y$):UNTIL a=h
2590 IF b=1 THEN dir$(k)="":RETURN ELSE dir$(k)=LEFT$(dir$(k),b):RETURN
2600 '_______ DATA
2610 '
2620 k=0:d$="":y$="/":fe$="":es=0:t=0:od=0:s=0:w=0:i=0:i0=0:i1=0:dn=0
2630 m_rwrec=&HE000:m_dlfat=&HE003:m_fatwt=&HE006:m_crsrw=&HE009
2640 m_devic=&HE00C:m_dir  =&HD000:m_dir2 =&HD006:m_msx1= &HD009
2650 m_trs  =&HD00F:m_var  =&HE01B:m_wttrc=&HE01E:m_sec00=&HE021
2660 m_dirwt=&HD003:m_rstor=&HE024:m_lddea=&HE018:m_saved=&HD015
2670 m_dldir=&HD012:m_clos2=&HD018:m_dir3 =&HD01B:m_dfree=&HD01E
2680 m_preop=&HD021:m_opens=&HD024:m_devo =&HD027:m_dirsb=&HD02A
2690 m_setdn=&HD02D:m_devi =&HD00C:m_gyoki=&HE045:m_tran =&HD030
2700 m_alta =&HD033:m_tranr=&HD036
2710 v_dn  =&HE08B:v_stop=&HE08C:v_ff  =&HE08D:v_bsiz=&HE08F
2720 v_bf  =&HE091:v_rec =&HE093:v_crs =&HE095:v_mac =&HE097
2730 v_pass=&HE098:v_edw =&HE099:v_iomm=&HE09A:v_edr =&HE09B
2740 v_iofg=&HE09C:v_msbt=&HE09D:v_slng=&HE09F:v_frwf=&HE0A1
2750 v_wfd0=&HE0A2:v_badr=&HE0A4:v_zoku=&HD080:v_fnam1=&HD084
2760 v_dirn=&HE051:v_morf=&HE052:v_flnm=&HE054:v_dpnf=&HE064
2770 v_sbdr=&HE065:v_wtnf=&HE066:v_bbaf=&HE067:v_p256=&HEE00
2780 v_fnam=&HE068:v_extn=&HE075:v_fbyt=&HE078:v_ecrs=&HE07C
2790 v_date=&HE07E:v_mmld=&HE083:v_mmjp=&HE085:v_fdop=&HE087
2800 v_od  =&HD180:v_op  =&HD181:v_fcrs=&HD182:v_yen =&HD184
2810 v_cdf =&HD185:v_fszl=&HD186:v_fszh=&HD188:v_ddrv=&HD18A
2820 v_bpth=&HE0A9:v_alpa=&HD060
2830 s_path=&HEC80:s_s=   &HECC0:v_csdir=&HDF00
2840 s_mac4   =s_s:s_tr4 =s_s+26:s_dn  =s_s+30:s_buff=s_s+31
2850 s_ff  =s_s+33:s_bsiz=s_s+35:s_escp=s_s+37:s_iomm=s_s+38
2860 RESTORE 2880:FOR i=1 TO 4:READ dms$(i):NEXT:POKE v_stop,0
2870 FOR i=1 TO 9:READ opt(i):NEXT:RETURN
2880 DATA X1(2HD),MS-DOS(2HD),X1(2D),MS-DOS(2D)
2890 DATA 0,1,2,3,&H10,&H11,&H12,&H13,0
2900 '________ ERROR
```


▶ふっ，５月号の奥津君！　まだまだ甘いな。私のディスプレイ(CZ-613D)には後ろの出力（音）端子から「貼り付けたトコがスピーカー」のダイヤトーンACT-1が付けられてあるのさ。
大西　力（18）香川県

```
2910 '
2920 DATA DEVICE I/O error
2930 DATA FAULT DISK error
2940 DATA COMMAND or FILE NAME error
2950 DATA DRIVE NUMBER error
2960 DATA DISK FULL
2970 DATA FILE ALREADY EXIST
2980 DATA Not WRITE
2990 DATA PATH error
3000 DATA Bad RECORD
3010 DATA Reserved future
3020 DATA Not EXEC
3030 '________ COMMAND DATA
3040 '
3050 DATA DIR,dir,TYPE,type,COPY,copy,DEL,del,CD,cd,end
```

リスト4

```
D000 C3 87 D7 C3 A5 D7 C3 2F : 52
D008 D8 C3 43 DA C3 7D D5 C3 : 90
D010 6E DD C3 0C DA C3 D3 D6 : 60
D018 C3 3C D7 C3 8C D7 C3 20 : DF
D020 DA C3 F3 D3 C3 CA D4 C3 : 87
D028 BB D5 C3 28 DC C3 86 DD : 7D
D030 C3 4A D2 C3 A0 D1 C3 D3 : A9
D038 D1 00 00 00 00 00 00 00 : D1
D040 00 00 00 00 00 00 00 00 : 00
D048 00 00 00 00 00 00 00 00 : 00
D050 00 00 00 00 00 00 00 00 : 00
D058 00 00 00 00 00 00 00 00 : 00
D060 00 00 00 00 00 00 00 00 : 00
D068 00 00 00 00 00 00 00 00 : 00
D070 00 00 00 00 00 00 00 00 : 00
D078 00 00 00 00 00 00 00 01 : 01
------------------------------------
SUM: F5 45 3C 2A 0D 4C 4B 5C B287

D080 00 02 04 00 00 00 00 00 : 06
D088 00 00 00 00 00 00 00 00 : 00
D090 00 00 00 00 00 00 00 00 : 00
D098 00 00 00 00 00 00 00 00 : 00
D0A0 00 00 00 00 00 00 00 00 : 00
D0A8 00 00 00 00 00 00 00 00 : 00
D0B0 00 00 42 49 4E 74 6F 41 : FD
D0B8 53 43 49 49 20 20 20 20 : A8
D0C0 20 20 9F 02 00 00 02 00 : E3
D0C8 85 36 20 18 23 00 00 00 : 16
D0D0 00 02 00 00 00 00 D0 20 : F2
D0D8 E0 00 03 02 00 00 C3 96 : 3E
D0E0 44 4F 53 32 20 20 20 20 : 98
D0E8 20 20 20 20 20 52 45 44 : 7B
D0F0 35 41 00 00 0F 00 00 00 : 85
D0F8 00 00 00 00 00 00 00 04 : 04
------------------------------------
SUM: 71 4D C4 00 E0 06 89 7F EF9D

D100 00 00 00 00 D8 00 E0 00 : B8
D108 03 81 00 00 C3 96 00 00 : DD
D110 00 00 00 00 00 00 00 00 : 00
D118 00 00 00 00 00 00 00 00 : 00
D120 00 00 00 00 00 00 00 00 : 00
D128 00 00 00 00 00 00 00 00 : 00
D130 00 00 00 00 00 00 00 00 : 00
D138 00 00 00 00 00 00 00 00 : 00
D140 00 00 00 00 00 00 00 00 : 00
D148 00 00 00 00 00 00 00 00 : 00
D150 00 00 00 00 00 00 00 00 : 00
D158 00 00 00 00 00 00 00 00 : 00
D160 00 00 00 00 00 00 00 00 : 00
D168 00 00 00 00 00 00 00 00 : 00
D170 00 00 00 00 00 00 00 00 : 00
D178 00 00 00 00 00 00 00 00 : 00
------------------------------------
SUM: 03 81 00 00 9B 96 E0 00 E90C

D180 02 01 3C 00 06 00 35 41 : BB
D188 00 00 07 07 01 07 20 00 : 36
D190 00 00 00 00 00 00 00 00 : 00
D198 00 00 00 00 00 00 00 00 : 00
D1A0 1A CD B1 D1 DC F7 D8 CD : E1
D1A8 B1 D1 D4 B8 D1 13 10 F0 : F2
D1B0 C9 FE 41 D8 FE 5B 3F C9 : 41
D1B8 C5 D6 41 4F 06 00 21 60 : B2
D1C0 D0 09 13 1A 1B FE 30 38 : 87
D1C8 07 FE 3A 30 03 D6 31 77 : F0
D1D0 34 C1 C9 AF 32 80 D1 32 : 22
D1D8 81 D1 CD 4A D2 D8 3A 97 : E4
D1E0 E0 FE 02 28 5E FE 04 28 : 90
D1E8 5A 3A 8B E0 FE 04 30 0E : 3F
D1F0 C5 D5 87 6F 26 00 11 FB : C2
D1F8 DD 01 02 00 18 1D FE 06 : 19
------------------------------------
SUM: C3 1A 43 71 74 B7 4C D6 F78F

D200 38 06 FE 16 38 3D D6 10 : AD
D208 C5 D5 21 05 00 D6 04 5F : F9
D210 16 00 CD 27 E0 11 03 DE : DC
D218 01 05 00 19 11 81 EE ED : 8C
D220 B0 21 01 EE 3A 00 EE B7 : 9F
D228 28 05 4F 06 00 ED B0 E1 : 00
D230 C1 48 06 00 ED B0 21 81 : 4E
D238 EE EB B7 ED 52 7D 32 80 : FE
D240 EE B7 C9 3E 0C 32 8C E0 : 56
D248 37 C9 21 80 EC 22 A9 E0 : 38
D250 C5 D5 CD 48 E0 D1 C1 D8 : F9
D258 C5 D5 3A 80 EE 47 11 81 : 1B
D260 EE CD 69 D2 D1 C1 38 E8 : A8
D268 C9 C5 D5 CD 21 D0 D1 C1 : B3
D270 D8 3A 84 D1 4F 78 91 EB : AA
D278 5F 16 00 19 EB 41 3E 01 : F9
------------------------------------
SUM: 38 45 AC 4B 94 75 9B 81 A70F

D280 32 65 E0 C5 D5 CD 24 D0 : D2
D288 D1 C1 C9 AF 21 00 EE 77 : 90
D290 23 22 F9 DD 3E 02 32 81 : 0E
D298 D1 32 65 E0 78 B7 28 05 : A4
D2A0 CD EF D3 28 05 CD 49 D3 : A5
D2A8 18 09 AF 32 51 E0 CD 93 : 93
D2B0 D3 13 05 CD C5 D3 38 50 : D8
D2B8 78 B7 28 5A D5 CD 77 D3 : 9D
D2C0 6F D1 0E 00 30 05 7C B7 : B6
D2C8 20 4C 0C C5 45 D5 CD 21 : 45
D2D0 D6 2A A4 E0 22 52 E0 D4 : AC
D2D8 00 D0 D1 C1 38 20 CD 20 : A7
D2E0 D3 21 51 E0 34 3E 04 BE : 59
D2E8 38 1E 2A 7C E0 22 95 E0 : 73
D2F0 CD 93 D3 CD 77 D3 38 1B : 9D
D2F8 4F 78 91 47 18 B3 3A 85 : 29
------------------------------------
SUM: B3 9D 24 88 0E 05 32 60 4B7A

D300 D1 B7 20 04 79 B7 20 0E : 0A
D308 78 32 84 D1 3E 08 32 8C : 03
D310 E0 37 C9 AF 18 05 AF 32 : 8D
D318 8C E0 78 32 84 D1 B7 C9 : EB
D320 D9 01 0D 00 ED 5B F9 DD : 05
D328 21 68 E0 ED B0 1B 1A FE : 39
D330 20 28 FA 13 3E 2F 12 13 : E7
D338 ED 53 F9 DD 21 01 EE EB : 11
D340 B7 ED 52 7D 32 00 EE D9 : 6C
D348 C9 D9 21 00 DF CD B9 DD : 05
D350 7E 32 51 E0 3D 87 4F 06 : FA
D358 00 3A 8B E0 5F 16 00 21 : 3B
D360 08 00 CD 27 E0 11 1A DF : E6
D368 19 09 5E 23 56 ED 53 95 : CE
D370 E0 ED 53 7C E0 D9 C9 C5 : E3
D378 0E 00 61 78 B7 28 0E CD : A1
------------------------------------
SUM: C9 0C F3 0E C9 A4 05 51 F1D4

D380 EF D3 28 0A FE 2A CC 91 : 79
D388 D3 13 0C 10 F2 37 79 C1 : 65
D390 C9 24 C9 3A 85 D1 B7 C8 : C5
D398 D9 21 00 DF CD B9 DD 3A : 76
D3A0 51 E0 77 D6 01 38 1C 87 : 5A
D3A8 4F 06 00 3A 8B E0 5F 16 : 6F
D3B0 00 21 08 00 CD 27 E0 11 : 0E
D3B8 1A DF 19 09 ED 5B 7C E0 : BF
D3C0 73 23 72 D9 C9 1A FE 2E : F0
D3C8 20 23 13 1A 1B FE 2E 20 : D7
D3D0 1C 3A 51 E0 D6 01 D8 D9 : 0F
D3D8 CD 51 D3 CD 93 D3 13 13 : 4A
D3E0 05 05 28 09 CD EF D3 28 : F2
D3E8 02 37 C9 13 05 B7 C9 1A : B4
D3F0 FE 2F C9 C5 CD 94 D4 21 : 11
D3F8 8A D1 CD 6C D7 C1 7E FE : A8
------------------------------------
SUM: 29 1E C5 39 4B 6C B5 7D B226

D400 07 C0 CD 2D D0 D8 C5 D5 : 03
D408 CD 76 D7 E5 DD E1 3A 80 : 77
D410 D1 D6 01 CE 00 6F 26 00 : 0B
D418 11 00 08 CD 27 E0 ED 5B : 35
D420 E1 EC 19 DD 75 23 DD 74 : AC
D428 24 DD E5 D5 22 8D E0 3A : 84
D430 97 E0 32 F8 DD CD 0C E0 : 37
D438 E1 DD E1 22 8D E0 D1 C1 : C0
D440 D8 CD 83 D4 3A 8B E0 DD : 7E
D448 77 27 2A A4 E0 DD 75 25 : C3
D450 DD 74 26 3A 97 E0 DD 77 : 7C
D458 28 DD E5 3A 81 D1 6F 3A : 1F
D460 65 E0 67 E5 CD 8B D2 E1 : 9C
D468 7C 32 65 E0 7D 32 81 D1 : F4
D470 DD E1 D8 3A 51 E0 DD 77 : 55
D478 2B 2A 7C E0 DD 75 2C DD : 0C
------------------------------------
SUM: 70 F4 96 44 7F 90 A9 B8 5E5E

D480 74 2D C9 3A F8 DD 21 97 : 31
D488 E0 BE C8 21 00 DF CD B9 : EC
D490 DD AF 77 C9 78 FE 03 C0 : 05
D498 AF 32 84 D1 21 40 EC 0E : 91
D4A0 00 D5 E5 06 03 1A BE 28 : C3
D4A8 06 CD F7 D8 BE 20 0F 13 : A2
D4B0 23 10 F2 E1 21 8A D1 51 : D3
D4B8 CD 6C D7 72 D1 C9 0C E1 : 09
D4C0 11 06 00 19 D1 7E B7 20 : 56
D4C8 D8 C9 CD 21 D6 D8 CD 76 : 80
D4D0 D7 E5 DD E1 DD 6E 23 DD : C5
D4D8 66 24 22 8D E0 DD 6E 25 : 89
D4E0 DD 66 26 22 52 E0 DD 7E : 18
D4E8 27 32 8B E0 DD 7E 28 32 : 79
D4F0 97 E0 CD 1B E0 CD 1E D0 : FA
D4F8 2A 95 E0 22 82 D1 DD 7E : 6F
------------------------------------
SUM: C1 CF 5B 0D 39 24 9C 21 7234

D500 2B 32 51 E0 DD 6E 2C DD : E2
D508 66 2D 22 95 E0 DD E5 CD : B9
D510 44 D5 DD E1 D8 CD 76 D7 : C9
D518 EB 21 68 E0 01 23 00 ED : 65
D520 B0 2A 7C E0 DD 75 29 DD : 8E
D528 74 2A DD 75 14 DD 74 15 : 6A
D530 2A 52 E0 DD 75 25 DD 74 : 24
D538 26 21 80 D0 CD 6C D7 3A : E1
D540 A3 E0 77 C9 3A 81 D1 FE : 4D
D548 03 DA 00 D0 28 05 CD 00 : A7
D550 D0 18 03 CD 1B D0 30 09 : DC
D558 3A 8C E0 FE 03 28 09 37 : 0F
D560 C9 3A 89 E0 B7 C2 42 E0 : 07
D568 AF 32 8C E0 21 01 00 22 : 91
D570 95 E0 CD 21 E0 2A 95 E0 : E2
D578 22 7C E0 B7 C9 3A 9C E0 : B4
------------------------------------
SUM: 13 42 8D 34 CA C3 22 0E 4F18

D580 3D 28 2D 3D 28 13 3E 01 : 49
D588 32 80 D1 2A A4 E0 22 91 : E4
D590 E0 FD 21 98 D5 C3 BB D5 : BE
D598 D8 CD 09 D0 3A 9C E0 FE : 32
D5A0 02 28 E3 3E 02 32 80 D1 : D0
D5A8 FD 21 AF D5 C3 BB D5 D8 : CD
D5B0 3A 9C E0 B7 20 E3 AF 32 : 51
D5B8 80 D1 C9 21 8A D1 CD 6C : CF
D5C0 D7 7E 87 4F 87 81 DD 21 : 31
D5C8 40 EC 4F 06 00 DD 09 DD : 44
D5D0 7E 03 B7 C8 DD 6E 04 DD : 2C
D5D8 66 05 FD E5 E9 3A 80 D1 : C1
D5E0 3D 32 A1 E0 CD 76 D7 E5 : EF
D5E8 DD E1 DD 6E 23 DD 66 24 : 93
D5F0 22 8D E0 DD 6E 29 DD 66 : 46
D5F8 2A 22 95 E0 DD 7E 28 32 : 76
------------------------------------
SUM: 41 5C E0 C7 D2 F3 78 F9 5ED3

D600 97 E0 DD 7E 27 32 8B E0 : 96
D608 DD E5 CD 1B E0 CD 09 E0 : 40
D610 DD E1 D8 2A 95 E0 DD 75 : 87
```

▶空飛ぶDNAデモを見て「パワードリフトでこんなコースがあったら面白いだろうなぁ」と思ったのは僕だけだろうか。
鴨居　大吾（18）香川県

```
D618 29 DD 74 2A AF 32 A1 E0 : 06
D620 C9 CD BE D6 78 B7 20 19 : 92
D628 3A 81 D1 B7 CA 36 E0 FE : 21
D630 02 CA 36 E0 FE 03 C8 3E : E9
D638 2A 32 C8 DD 32 D5 DD 18 : FD
D640 03 CD 7B D6 01 10 00 21 : 53
D648 C8 DD 3A 81 D1 FE 03 28 : 5A
D650 07 11 54 E0 ED B0 B7 C9 : 69
D658 06 0D 11 68 E0 CD 6D D6 : 7C
D660 06 03 21 D5 DD 11 75 E0 : 42
D668 CD 6D D6 B7 C9 7E FE 2A : 36
D670 C8 FE 3F 28 01 12 23 13 : 76
D678 10 F3 C9 D5 48 06 0D 21 : 1D
--------------------------------
SUM: 2C F6 9C 5F 4B 08 81 A8 2D57

D680 C8 DD 1A 13 FE 2E 28 19 : 3F
D688 77 23 0D 28 07 10 F3 CD : A6
D690 B4 D6 30 0D D1 3A 81 D1 : 24
D698 FE 02 C0 3E 2A 32 D5 DD : 0C
D6A0 C9 0D 28 0E 06 03 21 D5 : 0B
D6A8 DD 1A 77 13 23 0D 28 02 : DB
D6B0 10 F7 D1 C9 1A 13 FE 2E : FA
D6B8 C8 0D 20 F8 37 C9 C5 D5 : 87
D6C0 E5 01 0F 00 3E 20 11 C9 : 2D
D6C8 DD 21 C8 DD 77 ED B0 E1 : 98
D6D0 D1 C1 C9 CD 76 D7 E5 DD : 37
D6D8 E1 2A 86 D1 DD 75 10 DD : A1
D6E0 74 11 2A 88 D1 DD 75 12 : 6C
D6E8 DD 74 13 2A A4 E0 22 52 : 86
D6F0 E0 DD 7E 2B 32 51 E0 DD : A6
D6F8 6E 2C DD 66 2D 22 95 E0 : A1
--------------------------------
SUM: 82 9E 65 26 56 1F 3F F3 1BA7

D700 CD 1B D0 2A A4 E0 D4 36 : 70
D708 D7 22 52 E0 AF 32 8C E0 : 78
D710 CD D4 D9 DA 3C E0 CD 76 : B3
D718 D7 11 68 E0 01 23 00 ED : 41
D720 B0 21 80 D0 CD 6C D7 7E : AF
D728 32 A3 E0 3E 01 32 A1 E0 : A7
D730 CD 07 D9 C3 3C D7 CD 03 : 53
D738 E0 C3 12 D0 CD 76 D7 E5 : 84
D740 DD E1 DD 7E 2B 32 51 E0 : A7
D748 B7 28 11 DD 6E 2C DD 66 : AA
D750 2D 22 7C E0 E5 CD 03 E0 : 40
D758 E1 22 95 E0 DD 6E 23 DD : C3
D760 66 24 22 8D E0 CD 03 D0 : B9
D768 D4 06 E0 C9 C5 3A 80 D1 : D3
D770 4F 06 00 09 C1 C9 3A 80 : A2
D778 D1 6F 26 00 11 2E 00 CD : 72
--------------------------------
SUM: D3 9C D5 DF 39 97 5A B0 EC77

D780 27 E0 11 84 D0 19 C9 21 : 6F
D788 54 E0 18 03 21 68 E0 22 : DA
D790 EB DD CD B7 D7 AF 32 A1 : A5
D798 E0 CD DB D7 2A 91 E0 22 : 1C
D7A0 21 DE C3 2F D8 3E 01 32 : 3A
D7A8 A1 E0 CD DB D7 3E 20 32 : 90
D7B0 98 E0 AF 32 A1 E0 C9 C5 : 68
D7B8 D5 E5 2A EB DD 0E 02 06 : C2
D7C0 0D 7E FE 2A 28 05 23 10 : 13
D7C8 F8 18 06 3E 3F 77 23 10 : 3D
D7D0 FA 06 03 0D 20 EB B7 E1 : B3
D7D8 D1 C1 C9 2A A4 E0 22 91 : BC
D7E0 E0 3A 51 E0 B7 C2 09 E0 : AD
D7E8 C5 CD F8 D7 CD 00 E0 38 : 46
D7F0 05 0D 28 02 10 F6 C1 C9 : CC
D7F8 D5 E5 3A 97 E0 DD 21 34 : 9D
--------------------------------
SUM: C4 43 B5 2B BE 07 91 DC BF61

D800 E1 3D 28 12 DD 21 36 E1 : 6D
D808 3D 28 0B DD 21 38 E1 3D : C4
D810 28 04 DD 21 3A E1 DD 6E : 90
D818 00 26 00 22 93 E0 2A 8F : 74
D820 E0 ED 5B 9F E0 CD 2A E0 : 7E
D828 47 DD 4E 01 E1 D1 C9 CD : BB
D830 38 D8 CA 07 D9 C3 36 E0 : 93
D838 DD E5 CD 54 DD 06 80 2A : 70
D840 52 E0 ED 5B 21 DE B7 ED : 1D
D848 52 30 28 19 CD 12 E0 DD : 5F
D850 BE 00 28 1F DD BE 01 28 : C9
D858 14 C5 E5 CD A5 D8 CD B0 : 85
D860 D8 CC C2 D8 E1 E5 CC 7E : 4E
D868 D8 E1 C1 28 0B 11 20 00 : DE
D870 19 10 CF AF 3C 2A 52 E0 : 3F
D878 22 52 E0 DD E1 C9 DD 5E : 16
--------------------------------
SUM: E3 FA A4 19 BB F0 47 30 284E

D880 02 16 00 19 CD 12 E0 D6 : C6
D888 2E 20 02 3C C9 ED 5B EB : 88
D890 DD EB DD 46 03 CD E1 D8 : 74
D898 C0 2A EB DD 01 0D 00 09 : C9
D8A0 06 03 C3 E1 D8 DD 5E 04 : C4
D8A8 16 00 19 CD 12 E0 47 C9 : FE
D8B0 3A 65 E0 B7 C8 DD 4E 07 : 30
D8B8 3D 28 04 78 A1 A9 C9 78 : 6C
D8C0 A1 C9 3A 64 E0 B7 C8 FE : 65
D8C8 02 28 09 FE 03 28 0A 78 : DE
D8D0 DD A6 05 C0 78 DD A6 06 : 49
D8D8 C9 78 DD A6 06 DD AE 06 : 5B
D8E0 C9 7E FE 3F 28 0B CD 15 : 99
D8E8 E0 BE 28 05 CD F7 D8 BE : 25
D8F0 C0 23 13 10 EC AF C9 E5 : 4F
D8F8 67 E6 20 20 05 3E 20 B4 : A4
--------------------------------
SUM: 79 2F 08 91 34 A4 8C DC 4551

D900 18 03 3E DF A4 E1 C9 CD : 53
D908 96 D9 3A 97 E0 21 4C E1 : 6E
D910 3D 28 0F 21 72 E1 3D 28 : 4D
D918 09 21 4C E1 3D 28 03 21 : E0
D920 72 E1 46 23 C5 4E 06 00 : D5
D928 23 7E 23 5E 23 56 23 E5 : A3
D930 2A 52 E0 09 47 CD B5 D9 : 07
D938 E1 C1 10 E8 CD 4B D9 2A : B5
D940 52 E0 11 20 00 19 22 52 : F0
D948 E0 B7 C9 4E 23 06 00 EB : C2
D950 2A 52 E0 09 3A A1 E0 B7 : D7
D958 20 15 CD 12 E0 4F 21 87 : EB
D960 E0 06 04 1A 13 A1 77 23 : 52
D968 10 F9 79 32 A3 E0 C9 E5 : E5
D970 21 87 E0 EB 06 04 0E 00 : 8B
D978 1A B7 28 03 79 B6 4F 13 : 8D
--------------------------------
SUM: 3B D2 38 AD A1 11 CC 75 0DBD

D980 23 10 F5 D1 CD 8B D9 79 : A3
D988 C3 18 E0 3A A3 E0 B7 C8 : F7
D990 4F AF 32 A3 E0 C9 3A A1 : 57
D998 E0 B7 20 0B 21 68 E0 06 : 31
D9A0 23 AF 77 23 10 FC C9 ED : 2E
D9A8 5B 52 E0 06 20 AF CD 18 : 47
D9B0 E0 13 10 F9 C9 3A A1 E0 : 80
D9B8 B7 20 0C 79 FE 20 C4 12 : 50
D9C0 E0 12 23 13 10 F5 C9 EB : E1
D9C8 79 FE 20 7E C4 18 E0 23 : F4
D9D0 13 10 F5 C9 CD 54 DD 06 : E5
D9D8 00 2A 52 E0 11 20 00 CD : 5A
D9E0 12 E0 DD BE 00 28 0B DD : 9D
D9E8 BE 01 28 1C 19 10 F0 C3 : DF
D9F0 3C E0 E5 19 DD 7E 00 EB : 60
D9F8 CD 18 E0 EB 23 ED 5B A4 : BF
--------------------------------
SUM: 6F E5 EE 6C 33 C5 81 EF 379D

DA00 E0 B7 ED 52 22 9D E0 E1 : 56
DA08 22 52 E0 C9 CD 54 DD 2A : 45
DA10 52 E0 11 20 00 B7 ED 52 : 59
DA18 54 5D DD 7E 01 C3 18 E0 : C8
DA20 21 00 00 22 ED DD 23 22 : 52
DA28 95 E0 CD 21 E0 38 09 2A : AE
DA30 ED DD 23 22 ED DD 18 F2 : E3
DA38 AF 32 8C E0 2A ED DD 22 : 63
DA40 95 E0 C9 3A E5 EC E6 0F : 3E
DA48 CA 7A DB 3A 9C E0 3D CA : DC
DA50 CC DA 3D CA FD DA CD 3A : 8B
DA58 DB 3A EF DD FE 0D 28 3B : 4F
DA60 CD 15 E0 CD 58 DB 38 1C : 16
DA68 FE 0D 20 10 32 EF DD 3A : 73
DA70 E5 EC E6 0F FE 03 3E 0D : 12
DA78 20 0A 18 0A 21 8E D1 BE : 8A
--------------------------------
SUM: D0 BB 05 0F F9 58 1F 0C CC58

DA80 30 02 3E 20 13 0B D9 CD : 54
DA88 18 E0 13 0B D9 78 B1 CA : E2
DA90 E1 DA D9 78 B1 D9 CA B7 : 17
DA98 DA 18 BE AF 32 EF DD 3A : 97
DAA0 E5 EC E6 0F FE 03 38 04 : 03
DAA8 3E 0A 18 D8 CD 15 E0 FE : F8
DAB0 0A 20 AD 13 0B 18 D6 ED : D0
DAB8 53 F0 DD ED 43 F2 DD CD : EC
DAC0 22 DB AF 32 99 E0 3E 01 : 96
DAC8 32 9C E0 C9 ED 5B F0 DD : 8C
DAD0 ED 4B F2 DD D9 2A A4 E0 : 8E
DAD8 ED 4B 8F E0 09 EB D9 18 : 8C
DAE0 B8 D9 ED 53 F0 DD ED 43 : CE
DAE8 F2 DD D9 2A A4 E0 22 91 : 09
DAF0 E0 3A 9B E0 B7 20 1E 3E : C8
DAF8 02 32 9C E0 C9 CD AD DB : CE
--------------------------------
SUM: 3D 09 7D 2E 64 67 81 07 BDC8

DB00 ED 4B 9D E0 ED 5B A4 E0 : 81
DB08 D9 ED 5B F0 DD ED 4B F2 : 18
DB10 DD D9 C3 92 DA CD 22 DB : AF
DB18 3E 01 32 99 E0 AF 32 9C : 67
DB20 E0 C9 D9 2A A4 E0 ED 4B : 68
DB28 8F E0 09 22 91 E0 EB B7 : AD
DB30 ED 52 22 9D E0 CD 11 DC : 98
DB38 D9 C9 CD E0 DB AF 32 EF : FA
DB40 DD CD AD DB ED 4B 9D E0 : E7
DB48 ED 5B A4 E0 D9 2A A4 E0 : 53
DB50 ED 4B 8F E0 09 EB D9 C9 : 3D
DB58 21 E7 EC BE 28 0C 08 3A : 28
DB60 E5 EC E6 0F 3D 28 10 08 : 43
DB68 B7 C9 3E 01 32 9B E0 3A : A6
DB70 E8 EC 01 01 00 37 C9 08 : DE
DB78 37 C9 2A A4 E0 22 91 E0 : 41
--------------------------------
SUM: A9 9A D9 D2 BA 88 CA 03 A26D

DB80 3A 9C E0 3D 28 21 3D 28 : A1
DB88 0F CD E0 DB 18 0A AF 32 : 9A
DB90 99 E0 3E 01 32 9C E0 C9 : 2F
DB98 CD AD DB CD 11 DC 3A 9B : E4
DBA0 E0 B7 C2 18 DB 18 E7 3E : 89
DBA8 02 32 9C E0 C9 3A E5 EC : 84
DBB0 FE 10 D8 2A F4 DD 44 4D : 72
DBB8 ED 5B 8F E0 B7 ED 52 22 : CF
DBC0 F4 DD 2A F6 DD 11 00 00 : DF
DBC8 28 11 ED 52 22 F6 DD D0 : 3D
DBD0 3A 9B E0 B7 28 32 ED 43 : F6
DBD8 9D E0 C9 7C B5 C0 18 F0 : 3F
DBE0 21 00 00 22 86 D1 22 88 : 44
DBE8 D1 CD 76 D7 E5 DD E1 DD : 6B
DBF0 5E 10 DD 56 11 ED 53 F4 : E6
DBF8 DD DD 6E 12 DD 66 13 22 : B2
--------------------------------
SUM: 9C 6D 1F C4 07 B9 B3 D5 E3A5

DC00 F6 DD 7A B3 C0 7C B5 C0 : B1
DC08 3A E5 EC E6 0F 32 E5 EC : 03
DC10 C9 2A 86 D1 ED 5B 9D E0 : 0F
DC18 19 22 86 D1 2A 88 D1 11 : 26
DC20 00 00 ED 5A 22 88 D1 C9 : 8B
DC28 AF 32 84 D1 2A A4 E0 ED : D1
DC30 5B 8F E0 19 22 91 E0 EB : 61
DC38 3A 8C E0 B7 20 11 CD 21 : 7C
DC40 DD CD 81 DC D5 CD 06 D0 : 7F
DC48 D1 21 84 D1 34 18 E9 3A : B6
DC50 76 D0 B7 28 06 3E 0D CD : 43
DC58 18 E0 13 3E 1A CD 18 E0 : 28
DC60 13 2A 91 E0 EB B7 ED 52 : 8F
DC68 22 9D E0 AF 32 8C E0 3C : 28
DC70 32 99 E0 32 9C E0 3C 32 : C7
DC78 80 D1 FD 21 1C D6 C3 BB : DF
--------------------------------
SUM: 79 2A C0 2B 72 48 46 91 EAFC

DC80 D5 3A 76 D0 B7 20 0A CD : 03
DC88 98 DC 3E 0D CD 18 E0 13 : 97
DC90 C9 CD 0A DD 13 13 13 C9 : 7F
DC98 CD 0A DD 3A 8A E0 B7 28 : 37
DCA0 07 21 06 00 19 EB 18 0E : 58
DCA8 3A 7A E0 0E 00 CD E5 DC : 30
DCB0 2A 78 E0 CD D2 DC 13 CD : DD
DCB8 2E DD 3A 87 E0 E6 01 C8 : 5B
DCC0 3A 97 E0 FE 02 C8 FE 04 : 7B
DCC8 C8 13 2A 83 E0 0E 01 C3 : 3A
DCD0 D2 DC 7C CD E5 DC 7D CD : 02
DCD8 E5 DC 79 B7 C0 1B 3E 30 : 3A
DCE0 CD 18 E0 13 C9 F5 E6 F0 : 6C
DCE8 0F 0F 0F 0F CD F2 DC F1 : C8
DCF0 E6 0F B7 28 01 0C C6 30 : D7
DCF8 FE 3A 38 02 C6 07 CD 18 : 24
--------------------------------
SUM: 15 AF 78 A7 D0 6C D4 3D CC9B

DD00 E0 79 B7 3E 20 CC 18 E0 : 32
DD08 13 C9 21 68 E0 06 0D CD : 25
DD10 18 DD 13 06 03 C3 18 DD : C9
DD18 7E 23 CD 18 E0 13 10 F8 : 81
DD20 C9 D5 06 20 3E 20 CD 18 : 07
DD28 E0 13 10 F8 D1 C9 3A 89 : 58
DD30 E0 B7 3E 50 28 03 CD 18 : 35
DD38 E0 13 3A 8A E0 B7 21 C3 : 32
DD40 DD 20 0B 3A 87 E0 06 04 : B3
DD48 23 0F 38 02 10 FA 7E CD : C1
DD50 18 E0 13 C9 3A 97 E0 DD : 62
DD58 21 3C E1 3D C8 DD 21 44 : 85
DD60 E1 3D C8 DD 21 3C E1 3D : 3E
DD68 C8 DD 21 44 E1 C9 06 00 : BA
DD70 2A 91 E0 CD 7A DD 22 91 : 72
DD78 E0 C9 11 00 EE CD 12 E0 : 67
--------------------------------
SUM: DE B3 57 E6 FD 48 E2 9E 37B5

DD80 23 12 13 10 F8 C9 78 B7 : 48
DD88 C8 13 1A 1B FE 3A 20 27 : 8F
DD90 1A FE 61 38 02 D6 20 D6 : 7F
DD98 41 FE 1A 38 07 3E 04 32 : 0C
DDA0 8C E0 37 C9 32 8B E0 21 : 2A
DDA8 C0 EC C5 CD B9 DD C1 7E : 13
DDB0 32 97 E0 13 13 05 05 B7 : 90
DDB8 C9 C5 06 00 3A 8B E0 4F : 88
DDC0 09 C1 C9 44 4D 42 41 20 : C7
```

▶神様，風の神様，どーか私を神様にして……。つまりポピュラスが欲しいというわけです。
山梨　毅（17）神奈川県

```
DDC8 2A 20 20 20 20 20 20 20 : 0A
DDD0 20 20 20 20 20 2A 20 20 : 0A
DDD8 00 00 00 00 00 00 00 00 : 00
DDE0 00 00 00 00 00 00 00 00 : 00
DDE8 00 00 00 54 E0 3C 00 00 : 70
DDF0 00 00 00 00 35 F1 FF FF : 24
DDF8 03 01 EE 30 3A 31 3A 32 : F9
---------------------------------
SUM: E3 4B 81 4C 13 F9 FC 1C 8835

DE00 3A 33 3A 4D 45 4D 30 3A : F0
DE08 4D 45 4D 31 3A 45 4D 4D : 29
DE10 30 3A 45 4D 4D 31 3A 45 : F9
DE18 4D 4D 32 3A 45 4D 4D 33 : 18
DE20 3A 00 F0 00 00 00 00 00 : 2A
DE28 00 00 00 00 00 00 00 00 : 00
DE30 00 00 00 00 00 00 00 00 : 00
DE38 00 00 00 00 00 00 00 00 : 00
DE40 00 00 00 00 00 00 00 00 : 00
DE48 00 00 00 00 00 00 00 00 : 00
DE50 00 00 00 00 00 00 00 00 : 00
DE58 00 00 00 00 00 00 00 00 : 00
DE60 00 00 00 00 00 00 00 00 : 00
DE68 00 00 00 00 00 00 00 00 : 00
DE70 00 00 00 00 00 00 00 00 : 00
DE78 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 3E FF EE 05 11 10 04 FF 1611

DE80 00 00 00 00 00 00 00 00 : 00
DE88 00 00 00 00 00 00 00 00 : 00
DE90 00 00 00 00 00 00 00 00 : 00
DE98 00 00 00 00 00 00 00 00 : 00
DEA0 00 00 00 00 00 00 00 00 : 00
DEA8 00 00 00 00 00 00 00 00 : 00
DEB0 00 00 00 00 00 00 00 00 : 00
DEB8 00 00 00 00 00 00 00 00 : 00
DEC0 00 00 00 00 00 00 00 00 : 00
DEC8 00 00 00 00 00 00 00 00 : 00
DED0 00 00 00 00 00 00 00 00 : 00
DED8 00 00 00 00 00 00 00 00 : 00
DEE0 00 00 00 00 00 00 00 00 : 00
DEE8 00 00 00 00 00 00 00 00 : 00
DEF0 00 00 00 00 00 00 00 00 : 00
DEF8 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 00 00 00 00 00 00 00 00 0000

DF00 00 00 00 00 00 00 00 00 : 00
DF08 00 00 00 00 00 00 00 00 : 00
DF10 00 00 00 00 00 00 00 00 : 00
DF18 00 00 00 00 00 00 00 00 : 00
DF20 00 00 00 00 00 00 00 00 : 00
DF28 00 00 00 00 00 00 00 00 : 00
DF30 00 00 00 00 00 00 00 00 : 00
DF38 00 00 00 00 00 00 00 00 : 00
DF40 00 00 00 00 00 00 00 00 : 00
DF48 00 00 00 00 00 00 00 00 : 00
DF50 00 00 00 00 00 00 00 00 : 00
DF58 00 00 00 00 00 00 00 00 : 00
DF60 00 00 00 00 00 00 00 00 : 00
DF68 00 00 00 00 00 00 00 00 : 00
DF70 00 00 00 00 00 00 00 00 : 00
DF78 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 00 00 00 00 00 00 00 00 0000

DF80 00 00 00 00 00 00 00 00 : 00
DF88 00 00 00 00 00 00 00 00 : 00
DF90 00 00 00 00 00 00 00 00 : 00
DF98 00 00 00 00 00 00 00 00 : 00
DFA0 00 00 00 00 00 00 00 00 : 00
DFA8 00 00 00 00 00 00 00 00 : 00
DFB0 00 00 00 00 00 00 00 00 : 00
DFB8 00 00 00 00 00 00 00 00 : 00
DFC0 00 00 00 00 00 00 00 00 : 00
DFC8 00 00 00 00 00 00 00 00 : 00
DFD0 00 00 00 00 00 00 00 00 : 00
DFD8 00 00 00 00 00 00 00 00 : 00
DFE0 00 00 00 00 00 00 00 00 : 00
DFE8 00 00 00 00 00 00 00 00 : 00
DFF0 00 00 00 00 00 00 00 00 : 00
DFF8 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 00 00 00 00 00 00 00 00 0000

E000 C3 A6 E7 C3 61 E5 C3 F2 : 0E
E008 E5 C3 A1 E2 C3 00 E2 C3 : 93
E010 90 E6 C3 7A E6 C3 8A E6 : CC
E018 C3 96 E6 C3 82 E5 C3 3F : 6B
E020 E8 C3 6A E4 C3 28 E7 C3 : 8E
E028 63 E6 C3 44 E6 C3 B3 E6 : 92
E030 C3 D9 E6 C3 58 EA C3 65 : AF
E038 EA C3 69 EA C3 6D EA C3 : DD
E040 71 EA C3 75 EA C3 94 EA : BE
E048 C3 D8 EA 00 00 00 00 00 : 85
E050 00 00 C0 E0 3F 3F 3F 3F : 9C
E058 3F 3F 3F 3F 3F 3F 3F 3F : F8
E060 3F 3F 3F 3F 01 00 00 00 : FD
E068 44 4F 53 32 20 20 20 20 : 98
E070 20 20 20 20 20 52 45 44 : 7B
E078 35 41 00 00 0F 00 00 00 : 85
---------------------------------
SUM: 3E 1A 0B DC 08 82 B0 77 E6C3

E080 00 00 00 00 00 00 00 04 : 04
E088 00 00 00 00 00 00 D0 00 : D0
E090 10 00 F1 20 00 C3 96 03 : 7D
E098 20 01 02 02 01 AD 00 00 : D3
E0A0 01 00 03 04 00 E0 00 18 : 00
E0A8 1C 00 00 01 10 00 00 00 : 2D
E0B0 00 00 00 00 00 00 00 00 : 00
E0B8 00 00 00 00 00 00 00 00 : 00
E0C0 00 00 00 00 00 00 00 00 : 00
E0C8 00 00 00 00 00 00 00 00 : 00
E0D0 00 00 00 00 00 00 00 00 : 00
E0D8 00 00 00 00 00 00 00 00 : 00
E0E0 00 00 00 00 00 00 00 00 : 00
E0E8 00 00 00 00 00 00 00 00 : 00
E0F0 00 00 00 00 00 00 00 00 : 00
E0F8 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 4D 01 F6 27 11 50 66 1F 6CF2

E100 00 01 1A 10 00 10 1C 00 : 57
E108 1D 00 50 A3 0F 00 04 08 : 2B
E110 01 00 04 01 00 02 00 50 : 58
E118 CF 04 00 01 10 10 00 10 : 04
E120 0E 00 0E 00 28 FF 04 00 : 47
E128 02 09 02 00 04 01 00 02 : 14
E130 00 28 CF 02 20 10 05 06 : 34
E138 10 10 05 07 FF 00 01 0D : 39
E140 00 10 00 80 00 E5 00 08 : 7D
E148 0B 02 08 10 08 01 0D 68 : A3
E150 E0 0E 03 75 E0 11 01 98 : F0
E158 E0 12 02 78 E0 1E 02 7C : E8
E160 E0 18 05 7E E0 14 02 83 : F4
E168 E0 16 02 85 E0 00 07 10 : 74
E170 40 80 06 00 08 68 E0 20 : 36
E178 05 70 E0 08 03 75 E0 1C : D1
---------------------------------
SUM: DD 96 4C 46 FD 38 03 D0 7629

E180 04 78 E0 1A 02 7C E0 16 : EA
E188 04 7E E0 0B 00 02 01 10 : 80
E190 00 00 00 00 00 00 00 00 : 00
E198 00 00 00 00 00 00 00 00 : 00
E1A0 00 00 00 00 00 00 00 00 : 00
E1A8 00 00 00 00 00 00 00 00 : 00
E1B0 00 00 00 00 00 00 00 00 : 00
E1B8 00 00 00 00 00 00 00 00 : 00
E1C0 00 00 00 00 00 00 00 00 : 00
E1C8 00 00 00 00 00 00 00 00 : 00
E1D0 00 00 00 00 00 00 00 00 : 00
E1D8 00 00 00 00 00 00 00 00 : 00
E1E0 00 00 00 00 00 00 00 00 : 00
E1E8 00 00 00 00 00 00 00 00 : 00
E1F0 00 00 00 00 00 00 00 00 : 00
E1F8 00 00 00 00 00 00 00 00 : 00
---------------------------------
SUM: 08 F6 C0 25 02 7E E1 26 B38D

E200 C5 D5 E5 AF 32 A1 E0 CD : AE
E208 1A E2 38 0A 21 C0 EC CD : D8
E210 EE E6 3A 97 E0 77 E1 D1 : AE
E218 C1 C9 3A 8B E0 FE 04 D2 : 03
E220 81 E2 CD 28 E7 D8 CD 1B : FF
E228 E0 CD 21 E6 30 30 3A AB : F9
E230 E0 32 97 E0 CD 33 E0 CD : 36
E238 FC E6 3E 02 32 80 EB D4 : 93
E240 4D E7 D8 CD 24 E0 CD 1B : C5
E248 E0 CD 21 E6 D0 21 97 E0 : 1C
E250 34 3E 04 BE 30 DE 3E 03 : 83
E258 32 97 E0 C3 61 EA 3A 97 : 88
E260 E0 0E FE FE 02 28 06 0E : 28
E268 FD FE 04 20 12 2A 8D E0 : C8
E270 CD 7A E6 B9 20 B8 23 CD : AE
E278 7A E6 2B FE FF 20 AF B7 : 0E
---------------------------------
SUM: 82 22 44 D4 E1 84 C4 AB B03C

E280 C9 3E 03 32 97 E0 CD 1B : 9B
E288 E0 CD 21 E6 2A 8D E0 CD : 18
E290 7A E6 FE 01 20 08 23 CD : 77
E298 7A E6 FE 8F 28 E1 C3 61 : 1A
E2A0 EA C5 D5 E5 3A 8B E0 FE : 0C
E2A8 04 30 1E CD FC E6 38 1C : 55
E2B0 21 DA EC CD EE E6 3A 87 : 49
E2B8 EB 4F 7E C6 02 B9 38 01 : 72
E2C0 97 32 80 EB CD 4D E7 38 : 6D
E2C8 03 CD D0 E2 E1 D1 C1 C9 : BE
E2D0 AF 32 92 EB 3A 8A EB 32 : 3F
E2D8 8B EB B7 28 2D CD 20 E3 : 52
E2E0 CD 56 E3 38 35 3A 95 EB : 2D
E2E8 47 3A 96 EB B7 28 01 47 : 29
E2F0 CD 00 E0 38 25 21 92 EB : A8
E2F8 34 10 F5 3A 96 EB B7 20 : CB
---------------------------------
SUM: 80 B1 64 62 EB 49 AF 0B 2638

E300 09 3A 8B EB 3D 32 8B EB : 9E
E308 18 D0 3A 92 EB 6F 26 00 : 34
E310 ED 5B 9F E0 CD 63 E6 22 : FF
E318 9D E0 3E 00 32 96 EB C9 : 37
E320 2A 95 E0 3A 97 E0 FE 02 : 50
E328 28 1F FE 04 28 21 11 00 : A3
E330 01 B7 ED 52 38 09 19 11 : 62
E338 80 00 B7 ED 52 18 01 19 : A8
E340 29 29 29 29 22 93 E0 B7 : F0
E348 C9 11 09 00 19 18 F5 29 : 32
E350 11 08 00 19 18 EE 3A A1 : 13
E358 E0 B7 C2 75 E3 AF 32 96 : 28
E360 EB 32 9B E0 CD AE E3 3A : 30
E368 8F EB 32 96 EB 32 9B E0 : DA
E370 32 99 E0 B7 C9 2A 9D E0 : D2
E378 ED 5B 93 EB B7 ED 52 22 : DE
---------------------------------
SUM: FA BA 58 A9 DE FB 59 35 CB8D

E380 9D E0 28 02 30 1B 3A 99 : C5
E388 E0 B7 28 15 AF 32 99 E0 : 2E
E390 19 ED 5B 9F E0 CD 44 E6 : D7
E398 7C B5 28 01 1C 7B 32 96 : B9
E3A0 EB 2A 95 E0 22 90 EB CD : F4
E3A8 21 E0 D8 C3 DB E4 AF 32 : 3C
E3B0 8F EB 3A 97 E0 3D 28 09 : 99
E3B8 3D 28 33 3D CA 4F E4 18 : EA
E3C0 2D 2A 8D E0 ED 4B 95 E0 : 71
E3C8 09 CD 7A E6 5F 01 80 00 : 16
E3D0 09 CD 7A E6 57 ED 53 95 : 62
E3D8 E0 21 7F 00 B7 ED 52 D0 : 46
E3E0 21 8F 00 B7 ED 52 D8 7B : F9
E3E8 D6 7F 32 8F EB C9 2A 95 : 89
E3F0 E0 CB 3C CB 1D 08 E5 29 : E5
E3F8 D1 19 EB 2A 8D E0 19 08 : 8D
---------------------------------
SUM: B1 2D 06 15 5E BE A9 9B 7846

E400 38 0D CD 7A E6 4F 23 CD : B1
E408 7A E6 E6 0F 47 18 1D 23 : F4
E410 CD 7A E6 E6 F0 4F 23 CD : 42
E418 7A E6 47 B7 CB 18 CB 19 : 25
E420 CB 18 CB 19 CB 18 CB 19 : 8E
E428 CB 18 CB 19 ED 43 95 E0 : 6C
E430 78 B1 28 15 21 F7 0F B7 : 44
E438 ED 42 D0 3A 97 E0 FE 02 : B0
E440 28 07 79 D6 FD 32 8F EB : 27
E448 C9 3E 01 32 8F EB C9 2A : A7
E450 8D E0 ED 5B 95 E0 19 AF : F2
E458 32 96 E0 CD 7A E6 32 95 : 9C
E460 E0 FE 80 D8 D6 7F 32 8F : 4C
E468 EB C9 3A 97 E0 01 7A 01 : E1
E470 3D 28 0F 01 C7 04 3D 28 : A5
E478 09 01 50 00 3D 28 03 01 : C3
---------------------------------
SUM: B5 21 CE 47 AD 8F 2A 9A 258D

E480 63 01 3A 96 EB B7 28 22 : 20
E488 C6 7F 26 00 6F 3A 97 E0 : 8B
E490 FE 01 28 11 FE 03 28 0D : 6E
E498 21 FF 0F FE 02 28 06 3A : 97
E4A0 96 EB C6 FD 6F 22 95 E0 : 4A
E4A8 B7 C9 2A 95 E0 23 3A 97 : 13
E4B0 E0 FE 01 20 05 7D FE 80 : FF
E4B8 30 F3 C5 E5 22 95 E0 B7 : 1B
E4C0 ED 42 30 12 CD AE E3 E1 : B0
E4C8 C1 ED 5B 95 E0 7A B3 20 : CB
E4D0 DC 22 95 E0 B7 C9 E1 C1 : 95
E4D8 C3 3C E0 3A 97 E0 3D 28 : F5
E4E0 09 3D 28 21 3D CA 52 E5 : CD
E4E8 18 1B 2A 8D E0 ED 5B 90 : A2
E4F0 EB 19 EB 3A 95 E0 CD 96 : 01
E4F8 E6 EB 11 80 00 19 EB 3A : A0
---------------------------------
SUM: E4 0E 9B 65 7D F4 B3 26 8DBE

E500 96 E0 C3 96 E6 2A 90 EB : 5A
E508 CB 3C CB 1D 08 E5 29 D1 : D6
E510 19 EB 2A 8D E0 19 EB 08 : A7
E518 38 14 3A 95 E0 CD 96 E6 : 44
E520 13 CD 8A E6 E6 F0 47 3A : A7
E528 96 E0 B0 C3 96 E6 3A 96 : 35
E530 E0 47 3A 95 E0 B7 17 CB : 6F
E538 10 17 CB 10 17 CB 10 17 : 0B
E540 CB 10 4F 13 CD 8A E6 E6 : 60
E548 0F B1 CD 96 E6 13 78 C3 : 57
E550 96 E6 2A 8D E0 ED 5B 90 : EB
E558 EB 19 EB 3A 95 E0 C3 96 : F7
E560 E6 2A 7C E0 22 95 E0 22 : 25
E568 90 EB CD AE E3 2A 95 E0 : 78
E570 E5 21 00 00 22 95 E0 CD : 6A
```

▶姓を見ればわかると思うが，私こそ高麗人参飴をもらうにふさわしい人間だ。

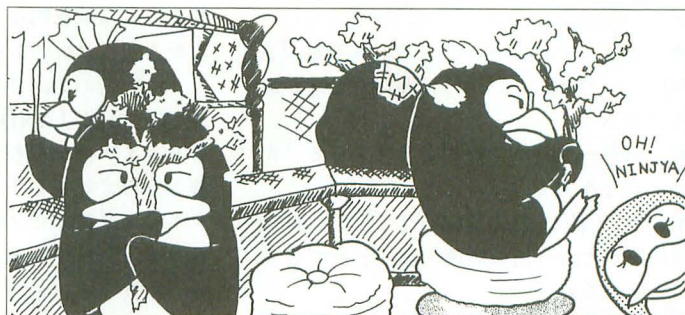
高麗　道也 (18) 徳島県

```
E578 DB E4 E1 3A 8F EB B7 28 : 33
--------------------------------
SUM: DC 00 8C 5B FF F6 6A 22 4A54

E580 E3 C9 C5 D5 E5 DD E5 3A : 27
E588 97 E0 DD 21 00 E1 3D 28 : BB
E590 12 DD 21 0D E1 3D 28 0B : 6E
E598 DD 21 1A E1 3D 28 04 DD : 3F
E5A0 21 27 E1 DD 6E 00 DD 66 : B7
E5A8 01 22 9F E0 DD 7E 02 32 : 31
E5B0 82 EB DD 7E 03 32 95 EB : 7D
E5B8 DD 5E 04 DD 56 05 ED 53 : B7
E5C0 93 EB 2A 8F E0 CD 44 E6 : 0E
E5C8 32 8A EB DD 6E 06 DD 66 : 3B
E5D0 07 22 83 EB DD 6E 08 DD : C7
E5D8 66 09 22 85 EB DD 7E 0A : 66
E5E0 32 87 EB DD 6E 0B DD 66 : 3D
E5E8 0C 22 88 EB DD E1 E1 D1 : 11
E5F0 C1 C9 3E 01 32 A1 E0 CD : 49
E5F8 21 E6 3A 97 E0 FE 01 28 : DF
--------------------------------
SUM: 3C 31 E3 38 1A 81 F5 7F CDED

E600 1B FE 03 28 17 21 03 00 : 7F
E608 22 83 EB 23 22 85 EB CD : 12
E610 21 E6 21 01 00 22 83 EB : B9
E618 23 22 85 EB AF 32 A1 E0 : 17
E620 C9 D5 E5 2A 8D E0 22 91 : CD
E628 E0 2A 83 EB 22 93 E0 CD : DA
E630 00 E0 38 0D 2A 85 EB ED : AC
E638 5B 93 E0 B7 ED 52 30 EF : E3
E640 B7 E1 D1 C9 3E 10 4B 42 : 0D
E648 EB 21 00 00 EB 29 EB ED : F8
E650 6A CA 5D E6 ED 42 30 04 : DA
E658 09 C3 5D E6 13 3D C2 4C : 6D
E660 E6 7B C9 C5 D5 3E 10 44 : 56
E668 4D 21 00 00 29 EB 29 EB : 96
E670 D2 74 E6 09 3D 20 F5 D1 : 58
E678 C1 C9 CD B3 E6 7E D2 D9 : 19
--------------------------------
SUM: 60 63 1B 26 F8 C3 57 2A 587F

E680 E6 C5 44 4D ED 78 C1 C3 : 25
E688 D9 E6 EB CD 7A E6 EB C9 : 8B
E690 EB CD 96 E6 EB C9 08 CD : BD
E698 B3 E6 38 05 08 12 C3 D9 : 8C
E6A0 E6 C5 08 42 4B ED 79 C1 : 67
E6A8 C3 D9 E6 C5 3A 8B E0 D6 : C2
E6B0 04 18 06 C5 3A 9A E0 D6 : 71
E6B8 01 3F DC BF E6 C1 C9 D6 : 21
E6C0 01 D8 20 0C 01 D0 1F 3A : 2F
E6C8 A2 E0 F6 10 ED 79 37 C9 : EE
E6D0 F3 3D 01 00 0B ED 79 B7 : 59
E6D8 C9 08 C5 01 D0 1F 3A A2 : 62
E6E0 E0 ED 79 01 00 0B 3E 10 : A0
E6E8 ED 79 C1 08 FB C9 06 00 : F9
E6F0 3A 8B E0 4F 09 C9 01 FF : C6
E6F8 0F ED 48 C9 01 FE 0F 3A : 55
--------------------------------
SUM: 80 2E 0B CE CD FC D6 1A 805F

E700 97 E0 FE 03 DA 08 E7 0C : 4D
E708 ED 78 3A 7F EB 87 87 87 : 9E
E710 87 C6 80 4F 3A 8B E0 81 : 42
E718 0E FC ED 79 C3 08 EA 01 : 26
E720 FC 0F 3A 8B E0 ED 79 C9 : DF
E728 C5 E5 CD FC E6 38 1B 01 : AD
E730 F8 0F 3A A6 E0 ED 79 CD : FA
E738 0D EA 38 0E 21 DA EC CD : F1
E740 EE E6 AF 77 32 80 EB CD : 64
E748 1F E7 E1 C1 C9 21 DA EC : 58
E750 CD EE E6 01 F9 0F 7E ED : 15
E758 79 3A A7 E0 57 3A 80 EB : 36
E760 CD 9A E7 D8 21 DA EC CD : DA
E768 EE E6 3A 80 EB 77 C9 21 : DA
E770 DA EC 06 00 3A 8B E0 4F : C0
E778 09 01 F9 0F 7E ED 79 3A : 30
--------------------------------
SUM: D0 69 5B 05 98 C1 02 81 B5AD

E780 80 EB BE C8 3A A8 E0 57 : 0A
E788 3A 80 EB CD 9A E7 D8 21 : EC
E790 DA EC CD EE E6 3A 80 EB : 0C
E798 77 C9 01 FB 0F ED 79 0E : BF
E7A0 F8 ED 51 C3 0D EA 3A 8B : B5
E7A8 E0 FE 04 D2 C5 E8 3A A1 : 3C
E7B0 E0 B7 20 14 2A 9F E0 2B : 9F
E7B8 22 0E EB 2A 91 E0 22 14 : EC
E7C0 EB 21 09 EB 3E 80 18 12 : E8
E7C8 2A 9F E0 2B 22 1E EB 2A : 29
E7D0 91 E0 22 1C EB 21 19 EB : BF
E7D8 3E A0 22 8D EB 32 8C EB : 21
E7E0 C5 D5 CD A9 E8 CD 78 E8 : 25
E7E8 D4 FC E6 D4 6F E7 38 3C : 54
E7F0 01 FA 0F 3A 81 EB ED 79 : 16
E7F8 0E F8 3A 8C EB ED 79 2A : 47
```

```
--------------------------------
SUM: 71 D3 00 53 4F 84 E5 B5 4171

E800 8D EB CD 88 EA CD 12 EA : 80
E808 38 22 3E 83 01 80 1F ED : A8
E810 79 2A 91 E0 ED 5B 9F E0 : DB
E818 19 22 91 E0 2A 93 E0 23 : 6C
E820 22 93 E0 01 FC 0F 3A 8B : 66
E828 E0 ED 79 B7 01 D0 1F 3A : 27
E830 A2 E0 ED 79 01 00 0B 3E : 32
E838 10 ED 79 FB D1 C1 C9 C5 : 91
E840 D5 2A 9F E0 2B 22 1E EB : D4
E848 2A 91 E0 22 1C EB CD A9 : 3A
E850 E8 CD 78 E8 D4 FC E6 D4 : 9F
E858 4D E7 38 16 21 19 EB CD : 74
E860 88 EA 01 F8 0F 3E F0 ED : 95
E868 79 CD 12 EA D4 80 EA D4 : 54
E870 1F E7 CD D9 E6 D1 C1 C9 : ED
E878 ED 5B 93 E0 2A 88 EB B7 : 0F
--------------------------------
SUM: 4C 0E 8E 92 00 14 1F 18 8D60

E880 ED 52 DA 79 EA EB 3A 82 : 23
E888 EB 87 5F 16 00 CD 44 E6 : DE
E890 32 80 EB 1E 01 3A 82 EB : 63
E898 67 7D 94 30 02 84 1D 3C : 87
E8A0 32 81 EB 7B 32 7F EB B7 : 6C
E8A8 C9 F3 CD B3 E6 30 0B 3E : 9B
E8B0 18 32 11 EB 3E 1C 32 20 : F2
E8B8 EB C9 3E 10 32 11 EB 3E : 6E
E8C0 14 32 20 EB C9 C5 D5 E5 : 99
E8C8 ED 5B 93 E0 2A 88 EB B7 : 0F
E8D0 ED 52 DA 79 EA 3A 8B E0 : 21
E8D8 D6 16 D2 9E E9 21 00 40 : A6
E8E0 22 97 EB 21 2B EB 11 06 : F2
E8E8 00 0E 00 CD AB E6 38 0B : AF
E8F0 E5 21 00 00 22 97 EB E1 : 8B
E8F8 0C 19 19 CD B3 E6 38 02 : DE
--------------------------------
SUM: 46 19 22 A3 E6 48 E7 92 6455

E900 0C 19 3A 9A E0 3D 47 3A : 97
E908 8B E0 D6 04 B8 28 0A 11 : 40
E910 43 EB 0D 28 04 CD D9 E6 : F3
E918 EB 11 1E 00 3A A1 E0 B7 : 8C
E920 28 01 19 11 51 E9 01 06 : 94
E928 00 ED B0 CD 36 E9 CD D9 : 2F
E930 E6 B7 E1 D1 C1 C9 2A 93 : 96
E938 E0 ED 5B 9F E0 CD 63 E6 : BD
E940 ED 5B 97 EB 19 44 4D D9 : 4D
E948 ED 4B 91 E0 D9 ED 5B 9F : 69
E950 E0 00 00 00 00 00 00 03 : E3
E958 D9 03 D9 1B 7A B3 20 F1 : 0E
E960 D9 ED 43 91 E0 D9 2A 93 : 10
E968 E0 23 22 93 E0 C9 CD B3 : E1
E970 E6 D9 CD 7A E9 D9 C9 CD : 5E
E978 AB E6 0A D2 D9 E6 ED 78 : 91
--------------------------------
SUM: 90 FF 7D 6A EC 80 DA 37 A288

E980 C3 D9 E6 08 CD B3 E6 D9 : C9
E988 CD 91 E9 D9 C9 08 CD AB : 69
E990 E6 38 05 08 02 C3 D9 E6 : AF
E998 08 ED 79 C3 D9 E6 87 87 : FE
E9A0 4F 06 0D 2A 93 E0 ED 5B : 47
E9A8 9F E0 CD 63 E6 ED 69 03 : EE
E9B0 ED 61 03 C5 2A 93 E0 11 : C4
E9B8 00 01 CD 44 E6 C1 ED 79 : 1F
E9C0 03 C5 D9 C1 D9 21 67 EB : AE
E9C8 3A A1 E0 B7 28 04 11 0C : BB
E9D0 00 19 CD B3 E6 38 04 11 : CC
E9D8 06 00 19 11 EB E9 01 06 : 0B
E9E0 00 ED B0 ED 4B 91 E0 ED : 33
E9E8 5B 9F E0 00 00 00 00 00 : DA
E9F0 00 03 1B 7A B3 20 F4 ED : 4C
E9F8 43 91 E0 2A 93 E0 23 22 : 96
--------------------------------
SUM: 3A 76 21 0F 5D 5C AA E3 AF1A

EA00 93 E0 CD D9 E6 C3 31 E9 : DC
EA08 16 81 C3 14 EA 16 99 C3 : CA
EA10 14 EA 16 FD 01 F8 0F 3A : 53
EA18 AC E0 5F 21 00 10 2B 7C : C3
EA20 B5 C2 29 EA 1D 20 F4 18 : D3
EA28 08 ED 78 A2 C2 1E EA B7 : 90
EA30 C9 ED 78 A2 4F E6 40 C4 : 09
EA38 75 EA 20 09 79 E6 81 C4 : 2C
EA40 5D EA CC 61 EA 3A 8B E0 : 03
EA48 01 FC 0F ED 79 CD 80 EA : A9
EA50 21 00 00 22 95 E0 37 C9 : B8
EA58 AF 32 8C E0 C9 3E 01 18 : 6D
EA60 1A 3E 02 18 16 3E 03 18 : E1
EA68 12 3E 04 18 0E 3E 05 18 : D5
EA70 0A 3E 06 18 06 3E 07 18 : C9
EA78 02 3E 09 32 8C E0 37 C9 : E7
--------------------------------
```

```
SUM: CA C1 BA 0C EF AA 2C 75 FBE7

EA80 3E 83 01 80 1F ED 79 C9 : 90
EA88 01 80 1F 5E 23 04 ED A3 : B5
EA90 1D 20 FA C9 01 00 FE ED : EC
EA98 5B 9D E0 2A 91 E0 DD 21 : 71
EAA0 02 EE 7A B3 20 0C 3A E7 : 6A
EAA8 EC DD BE FF 20 01 0D AF : 63
EAB0 18 17 1B CD 7A E6 23 FE : 98
EAB8 0D 20 04 3E 02 18 0A DD : 70
EAC0 77 00 DD 23 0C 10 DB 3E : AC
EAC8 01 32 00 EE 79 32 01 EE : BB
EAD0 22 91 E0 ED 53 9D E0 C9 : 19
EAD8 2A A9 E0 DD 21 81 EE AF : CF
EAE0 32 80 EE 4F 7E 23 B7 CA : 11
EAE8 65 EA FE 3B 28 08 DD 77 : 0C
EAF0 00 DD 23 0C 18 EE 1A 13 : 3F
EAF8 DD 77 00 DD 23 0C 10 F6 : 66
--------------------------------
SUM: 02 EC FD DC 6A 61 1D D9 5898

EB00 79 32 80 EE 22 A9 E0 B7 : 7B
EB08 C9 0F 83 7D FB 0F FF 00 : E1
EB10 2C 18 80 8D 00 EF 92 CF : A1
EB18 87 11 83 79 00 D8 FF 00 : 6B
EB20 1C 28 80 8D FB 0F 92 CF : BC
EB28 05 CF 87 ED 78 D9 ED 79 : FF
EB30 D9 ED 78 D9 02 00 D9 0A : FC
EB38 00 D9 ED 79 D9 0A 00 D9 : FB
EB40 02 00 D9 CD 77 E9 CD 83 : 58
EB48 E9 D9 ED 78 D9 ED 79 D9 : 3F
EB50 0A 00 D9 ED 79 D9 ED 78 : 87
EB58 D9 02 00 D9 0A 00 D9 02 : 99
EB60 00 CD 6E E9 CD 8D E9 D9 : 40
EB68 ED 78 D9 ED 79 D9 ED 78 : E2
EB70 D9 02 00 ED 78 D9 ED 79 : 7F
EB78 D9 0A 00 D9 ED 79 D9 01 : FC
--------------------------------
SUM: 5C 53 58 E4 E9 D8 70 52 69E3

EB80 00 10 10 0E 00 0E 00 28 : 64
EB88 FF 04 01 01 80 09 EB 00 : 79
EB90 13 00 02 00 10 10 00 00 : 35
EB98 00 00 00 00 00 00 00 00 : 00
EBA0 00 00 00 00 00 00 00 00 : 00
EBA8 00 00 00 00 00 00 00 00 : 00
EBB0 00 00 00 00 00 00 00 00 : 00
EBB8 00 00 00 00 00 00 00 00 : 00
EBC0 00 00 00 00 00 00 00 00 : 00
EBC8 00 00 00 00 00 00 00 00 : 00
EBD0 00 00 00 00 00 00 00 00 : 00
EBD8 00 00 00 00 00 00 00 00 : 00
EBE0 00 00 00 00 00 00 00 00 : 00
EBE8 00 00 00 00 00 00 00 00 : 00
EBF0 00 00 00 00 00 00 00 00 : 00
EBF8 00 00 00 00 00 00 00 00 : 00
--------------------------------
SUM: 12 14 13 0F 90 27 EB 28 7B9A

EC00 00 00 00 00 00 00 00 00 : 00
EC08 00 00 00 00 00 00 00 00 : 00
EC10 00 00 00 00 00 00 00 00 : 00
EC18 00 00 00 00 00 00 00 00 : 00
EC20 00 00 00 00 00 00 00 00 : 00
EC28 00 00 00 00 00 00 00 00 : 00
EC30 00 00 00 00 00 00 00 00 : 00
EC38 00 00 00 00 00 00 00 00 : 00
EC40 4E 55 4C 01 1C D6 43 4F : 74
EC48 4E 00 00 00 41 55 58 00 : 3C
EC50 00 00 50 52 4E 00 00 00 : F0
EC58 24 31 40 00 00 00 24 32 : EB
EC60 40 00 00 00 24 33 40 00 : D7
EC68 00 00 00 00 00 01 DD D5 : B3
EC70 00 00 00 00 00 00 00 00 : 00
EC78 00 00 00 00 00 00 00 00 : 00
--------------------------------
SUM: 00 86 DC 53 CF 5F DC 56 C0C8

EC80 65 3A 2F 3B 66 3A 2F 3B : 13
EC88 77 3A 2F 3B 61 3A 2F 3B : 20
EC90 62 3A 2F 3B 00 00 00 00 : 06
EC98 00 00 00 00 00 00 00 00 : 00
ECA0 00 00 00 00 00 00 00 00 : 00
ECA8 00 00 00 00 00 00 00 00 : 00
ECB0 00 00 00 00 00 00 00 00 : 00
ECB8 00 00 00 00 00 00 00 00 : 00
ECC0 03 01 03 03 03 03 03 03 : 16
ECC8 03 03 03 03 03 03 03 03 : 18
ECD0 03 03 03 03 03 03 03 03 : 18
ECD8 03 03 00 00 00 00 00 00 : 06
ECE0 E0 00 D0 00 10 10 02 1A : EC
ECE8 1A 00 00 00 00 00 00 00 : 1A
ECF0 00 00 00 00 00 00 00 00 : 00
ECF8 00 00 00 00 00 00 00 00 : 00
--------------------------------
SUM: 44 B8 66 BA E0 8D 69 99 576B
```

マシン語カクテル in Z80's Bar
第12回――帰ってきたゼンジソフト―

シナリオ：西川善司
特別監修：金子俊一
イラスト：山田純二

ひさびさに主役で登場の善ちゃんがまたまた重ね合わせのアルゴリズムに迫る。ぼっくんな性格とは裏腹にゼンジソフトって結構まともなプログラムを作るんですよねー。まあ，本人の実力か，長老や光君の手助けのおかげかどうかは知らないけど……。

♪カラン，コローン（ドアが開く音）

西川善司(以下善)：突然ですが長老。今我がゼンジソフトでは第4回作品を制作中なんだ……。

ようこ(以下Yo)：また『なになにの仕方がわからないんですよ』なんて言うんでしょ。

源光(以下光)：まったく，それでよくぬけぬけと『我がゼンジソフトでは……』なんて言えるよな。

長老(以下老)：まぁ，よい。それで？

善：えぇ，キャラクタの表示の仕方なんだけど……。

Yo：あれぇ？　西川さんさあ，以前3重スクロールの話（'89年11月号）のときに女の子のキャラクタをグラフィックで表示してたじゃない？

善：あぁ，あれね。今回はあれと違ったのやりたいんだよ。

光：いったい何をやりたいんだ？

マスター(以下M)：まさかイースとかみたいにキャラクタとキャラクタを重ね合わせたいとか？

善：そのと一り。

M：……（なに威張ってんでしょうね，この男は)。

まずは復習から

老：まず初めにグラフィックにキャラクタを表示するにはどうしたらいいかを復習してみようか。

Yo：えぇ，確かグラフィックRAM(以下G-RAM)にデータを書き込むんだったわよね。

光：そのとおり。X1の場合，PCG用データフォーマット(注1)のデータ群をG-RAMに書き込むというのが一般的な手法として挙げられるね。

善：そう，あの3重スクロールの話のときにも使ったやつだな。

M：これですね（リスト1)。

老：まず，6行目はDレジスタにループカウンタを代入している。「8＊3」の「8」は1キャラクタが縦8ドットで構成されていることによる。「8＊3」の「3」のほうは1キャラクタがRGB（赤緑青）3つのプレーンから構成されていることを意味しているのじゃ。

Yo：8，9行はHLレジスタが指し示すアドレスの内容をI/O（この場合はG-RAM）へ転送して，しかもHLレジスタを自動的に1だけ増加してくれるんだったわよね。

光：そうだね。よく「INC B」と「OUTI」は対になって用いられるね。

善：10～12行では次に表示するG-RAMのアドレスを処理しているんだ。

老：10～12行はとても重要だぞ。図1にBCレジスタの変化の様子を示しておくから初心者はよく見て理解しよう。

Yo：13行から下は単純なループ制御ね。

ANDとORであんどーとわー

善：そういや，最近メアリーはこの店に来てないんですか。先月も5月だというのに春休みとかいって休んでたみたいだし。まったくいい加減な奴だなー。

メアリー(以下メ)：Zen!　Whoガイイ加減デスカ。

老：おお！　これぞ，『壁に耳あり，障子に』

一同：メアリー！(目あり)

M：まぁ，冗談はそれくらいにして……。

善：そもそも，なんでX1であんな重ね合わせ処理ができるんだろう，スプライトもないのに。X1に特殊な隠れ機能でもあるのかねぇ。

光：はっはっは。自分に理解できないことを，すぐそういうふうに片づける奴がよくいるんだよな。

善：なんだてめー。じゃ，お前はわかるのかよぉ。

光：ははは。わかるとも。マスク処理というのをしてやるんだ。

老：まあまあまあ，仲よくやろうじゃないか。確かに光のいうとおり「マスク処理」というのを用いてやるんじゃ。

リスト1

```
 1 ;       1キャラクタ表示ルーチン
 2 ;ENTRY
 3 ;    BC=G-RAM ADDR
 4 ;    HL=CHR DATA ADDR
 5
 6         LD     D,8*3
 7 LOOP:
 8         INC    B
 9         OUTI
10         LD     A,B
11         ADD    A,08H
12         LD     B,A
13         DEC    D
14         JR     NZ,LOOP
15         RET
```

図1　アドレス加算の様子

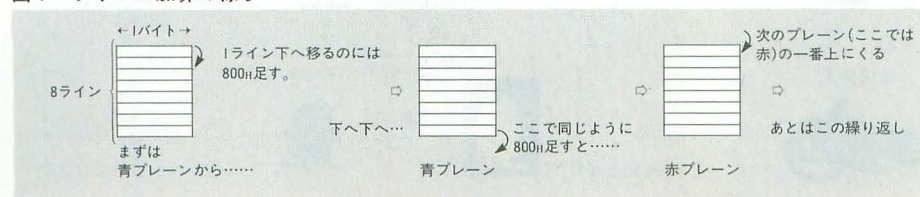



図2　ANDとORの真理値表

AND

X	Y	X AND Y
0	0	0
0	1	0
1	0	0
1	1	1

OR

X	Y	X OR Y
0	0	0
0	1	1
1	0	1
1	1	1

メ："Mask" means「隠ス，覆ウ」ネ。

老：そう，それじゃ。ところでみんな，「AND」と「OR」というのは聞いたことがあるじゃろう。

善：スペースハリアーで1ミスすると「オァー」なんちて，ぽっくん。

Yo：(軽く流して)あ，知ってる。大学の「論理回路」とか「情報処理理論」とかでやったことあるわ。

M：BASICの「IF」文とかにもあるよね。

光：ここで言っているのはようこさんのほうの論理演算の「AND」と「OR」です。

老：さて，この「AND」と「OR」の真理値表はこうなるのじゃが(図2)，なにかこれを見て気づいたことはないかな。

図3 ANDとORの図形的イメージ

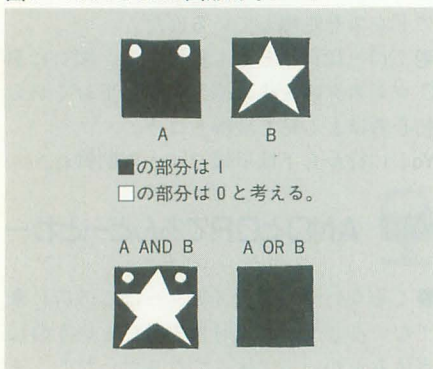
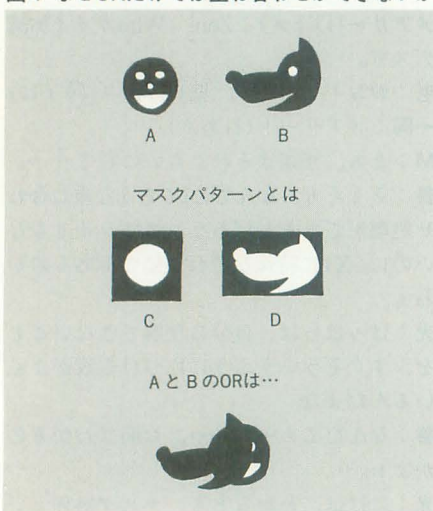
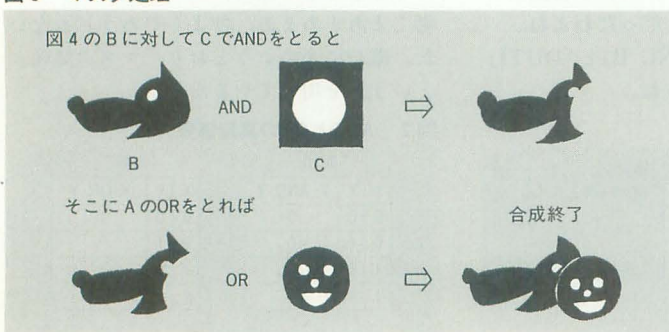



図4 なぜORだけでは重ね合わせができないか

善：そんな抽象的なこと言われたってねぇ，わかるわけないじゃん。

……2，3分後……

メ：Oh！ 私「AND」ノ方，分カッタ。Xヲ0ト1ニ分ケテ考エルトイイネ。

善：分けて考える？

メ：ツマリXガ0ノ時ニハYニ何ガ来テモ0ニナッテシマッテルネ。

Yo：あ，なるほど。

メ：Xガ1ノ時ニハYニ来タ値ガ，ソノママ演算ノ結果ニナッテルネ。

善：ほぇ～。んだからなにー。それがどしたのー。(←結構やな奴)

老：いやいや，御名答，御名答。Yを入力データと考えるとXを0にしたときは，Yが何になろうが0としていることから，データの「切り捨て」と考えられる。それに対して……。

光：Xを1にしたときはYをそのまま演算結果としていることからデータの「保存」と考えることができる。

M：なるほどね。

善：「OR」のほうも同じようにして考えると……。うー。わけわからん。単に「OR」のほうはXかYどっちかが1なら演算結果も1，XとYの両方0のときに演算結果を0としているだけで特に気づく点というのはないけどなぁ……。

老：それ，それじゃよ。わしが聞きたかったのは。つまり，「OR」は2つのデータの合成と考えることができるじゃろう？

Yo：あ，ほんとだ。

メ：私ハ初メカラワカッテマシタ。

善：あ，わかった。重ね合わせ処理には「OR」を使うんでしょ。(興奮気味)

光：ははは。早とちりな奴だな。確かに「OR」も使うが「OR」だけでは完全な重ね合わせ処理にはならないのだ。

老：まぁ，光よ。順を追って説明しとるんじゃから先走るでない。さて，みんな，この「AND」と「OR」の真理値表を図形的なイメージで表すといっそうわかりやすくなるぞ。

マスク処理とは

老：たとえば，図3のAという図形とBという図形の「AND」や「OR」をとったらどうなるか考えてごらん。

メ：コウネ(同じく図3)。

善：うん，なるほど，わかる，わかる。しかし，なんで重ね合わせをするのに「OR」だけじゃ駄目なんだよ。

光：それじゃ，教えてやろう。いいか，図4を見てみろ。人の顔の絵（A）と犬の顔(B)の絵があったとするぞ。このAとBをORで重ね合わせた場合を考えてみろ。

善：あ，確かに。目や口や鼻とかの空白の部分が上から重ね書きされてしまって塗り潰されちゃうわけね。

光：そこで「マスク処理」が必要になってくるわけだ。マスク処理を行うにはマスクパターンを用意しなくてはならない。

Yo：マスクパターン？

光：そう，マスクパターンとは今の図4でいくとCやDのことだな。

メ：How to make マスクパターン？

老：まあ，それはわしがあとで説明するとしよう。ここでは，まず，マスク処理とはどんなものかを理解するのじゃ。

M：で，そのマスクパターンをどうするのですか？

光：すでに描いてある絵に，次に描く絵のマスクパターンでANDをとってやるんです。

善：んー。そのマスクパターンとやらでANDをとると……。あ，なるほど，こうなるわけか(図5)。

Yo：そこに次から描きたい絵でORしてやると合成終了ってわけね。なるほど。

老：さて，マスクパターンの作り方じゃが……。

M：単純に元絵の反転画ではないようですね。

善：それに，実際はRGB 3プレーンで絵が作られているわけだし。

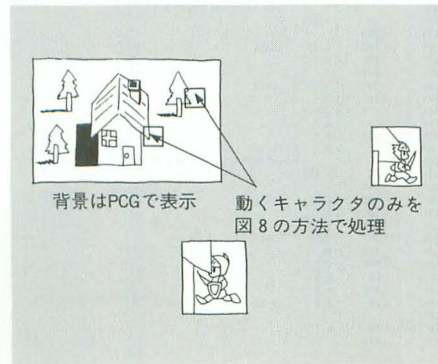
老：さっき西川が「OR」だけを用いて重ね

図5 マスク処理

図6 マスクパターンの作り方

合わせをやって失敗したじゃろ。逆に実はマスクパターンを作るにはその方法を用いるんじゃ。
M：というと……？
老：図６を見てもらえば簡単にわかるじゃろう。まず，RGB３プレーン分の絵をすべてORで重ねてしまう。そして次に，その絵の反転処理をして出来上がりじゃ。
メ：思ッタヨリ簡単ニ出来ルネー。
善：最初にやったマスク処理の例は１プレーン分のやつだったよね。RGB３プレーン分のマスク処理はどうやるの？
光：重ねられる絵のほうもRGB３プレーン，これから重ねる絵のほうも３プレーンあるとする。マスクパターンで，重ねられる側の絵のRGB３プレーンをマスクしてあとは合成して出来上がりだ(図７)。

マスク処理の応用

Yo：なるほどね。今日は勉強になったわぁ。
善：あ，ちょっと待って。みんな帰り支度しないように。
老：まだ何かあるのか？
善：うん。ち。なんちて，ぽっくん。
光：さ，帰ろうか，みんな。
善：うわー。ちっと待ってよぉ。あの，イースなんかでさ，自キャラが木の後ろに隠れたりするじゃん。あいうのってどうやるの？
メ：Oh！　オーバー処理ノコトネ。
Yo：あれ，それってやっぱりこの間（'89年11月号）やらなかったっけ？
善：あれはキャラクタ単位のオーバー処理だったでしょ。仮想画面を見て，あるキャラクタ（文字）が自分と重なったときに「描く/描かない」で処理してやる，比較的単純なものなんだよ。ソーサリアンなんかはこの方法だね。でも，イースやなんかだと小枝の隙間からキャラクタが見え隠れするじゃん。
老：実はこれもマスク処理の応用でできることなんじゃよ。では木の後ろにキャラクタが隠れるという場合を想定しようか。まず，地面と自キャラをマスク処理で合成する。その絵に木の絵を合成して出来上がりじゃ(図８)。
メ：トイウコトハ，地面以外ノキャラクタ(この場合は人間と木)ニハ，マスクパターンガ必要ナワケネー。
M：え？　じゃ，人間が重ならない場合も木と地面をいつも合成処理しなくちゃいけないんですか。なんか凄く処理が重くなりそうですよ。
光：いやいや。だから，木と地面が重なり合ったような絵はあらかじめ用意しておいて普段はそれを表示しておき，人間などのキャラクタと重なったときだけ今のような処理をするのさ。
善：つまり，そういう背景なんかはX1だとPCGなんかで表示しておいて，動くキャラクタなんかのみをさっきのような処理でグラフィック表示してやるわけだね(図９)。
老：そのとおりじゃ。PCGはテキストと同じじゃからスクロールなどの処理も高速に行えるしな。

サンプルプログラム

……数時間後……

善：こんなもんでどうでしょうかね。
光：なにこれ。ただのサブルーチンじゃないか。
善：はははは。
M：なにが「はははは」ですか。
老：まあ，よい。それで？　このルーチンを説明してもらおうか。
善：リストの初めにコール例を書いておいたからそれを読んでもらえればわかるとは思うんだけど，まず，MAKE_MASKというルーチンはマスクパターンを作ってくれる。パラメータはIX，IY，HLの３つのレジスタ。IXには元のキャラクタデータの格納アドレスを入れ，IYにはマスクパターン

図７　RGB３プレーンの場合のマスク処理

図８　マスク処理の応用

図９　Ysなんかの場合

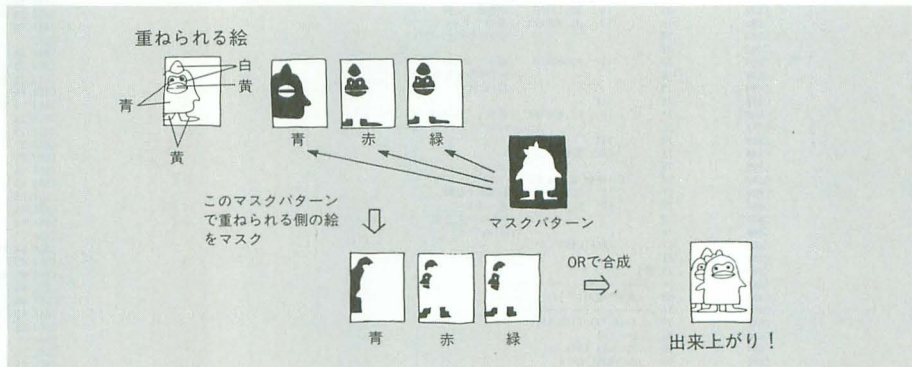
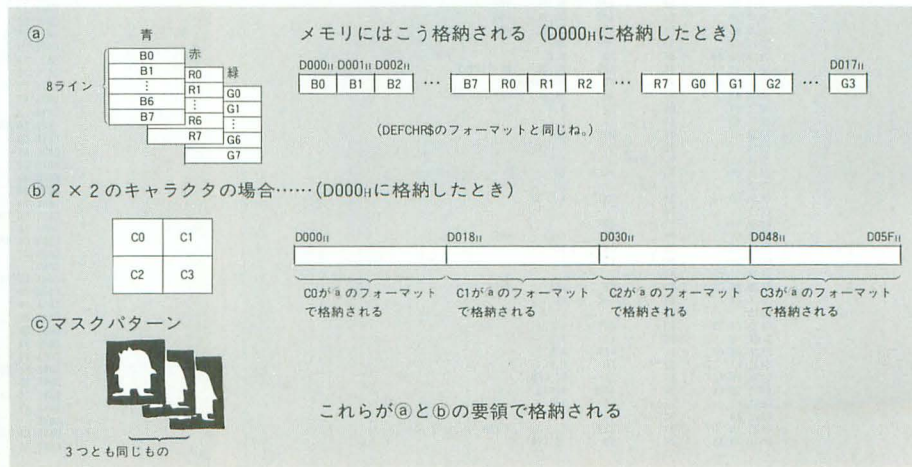



図10　サンプルプログラムのマスクパターンデータとキャラクタデータのフォーマット

を格納するアドレスを入れる。

Yo：HLには？

善：HL にはマスクパターンを作りたいキャラクタの数×8を入れる。3つのレジスタに適当な値を入れたらあとはCALLしてやるだけだ。

老：キャラクタデータはどういうフォーマットで格納しておくのかな。

M：それにマスクパターンはどういうふうに格納されるんですか。

善：図10のような感じ。

メ：マスクパターンハ，ナゼ同ジモノガ3ツモツナガッテ格納サレルノデスカー？

善：それはね，3プレーン分マスクを取るときにいちいちマスクパターンをポイントしているポインタレジスタを初期化するのが面倒だからさ。簡単にいえば高速化をしたらこうなった，てな感じだな。

Yo：よくある話ね。

善：次が実際にキャラクタを PUT するルーチン。パラメータとなるレジスタはBC, HL, DE, HL'の4つ。BC のBにY座標（0～49），CにX座標（0～79）を入れる。HL には表示するキャラクタの格納アドレスを入れる。DE レジスタには表示するキャラクタの大きさを入れてやる。DにはX方向の長さ，EにはY方向の長さだ。

老：HL' にはMAKE_MASKで作ったマスクパターンの格納アドレスを入れるわけじゃな。

M：それでCALLですね。

善：最後のは画面に描いたキャラクタを消すルーチンだ。これは表示したときと同じ要領でBC座標，DEに大きさを入れて CALLしてやればOK。

メ：HLヤHL'ハ，イラナイノネー。

光：まぁ，実際には表示するたびにマスクパターンを作っていたのでは処理が遅くなるのであらかじめメモリに展開しておくのがいいだろうね。

老：また，アクションゲームなどを作る際には，表示プレーン数を少なくしたり，表示キャラクタの大きさごとに専用の表示ルーチンを作ってやるなどして高速化しなくてはいかんじゃろうな。

―つづく―

注1）いわゆるDEFCHR$のフォーマットのこと。図10-a参照

リスト2

```
C800            1           ORG     0C800H
C800            2       ;コール例 1 (ﾏｽｸﾊﾟﾀｰﾝを作くる)
C800            3       ;   LD   IX,0D000H  ;元のｷｬﾗｸﾀﾃﾞｰﾀ
C800            4
C800            5       ;   LD   IY,0D600H  ;作ったﾏｽｸﾊﾟﾀｰﾝ
C800            6                           ;のﾊﾟ格納ｱﾄﾞﾚｽ
C800            7
C800            8       ;   LD   HL,64*8    ;ｶｳﾝﾀ=ｷｬﾗｸﾀ数x8
C800            9       ;   CALL MAKE_MASK
C800           10
C800           11
C800           12       ;コール例 2 (表示する)
C800           13       ;   LD   BC,1008H   ;X=08H,Y=10H
C800           14       ;   LD   HL,0D000H  ;表示したい
C800           15                           ;ｷｬﾗｸﾀのﾃﾞｰﾀ･ｱﾄﾞﾚｽ
C800           16
C800           17       ;   LD   DE,0402H   ;表示したい
C800           18                           ;ｷｬﾗｸﾀの大きさ
C800           19                           ;XL=4,YL=2
C800           20       ;   EXX
C800           21       ;   LD   HL,0D600H  ;表示したいｷｬﾗｸﾀの
C800           22                           ;ﾏｽｸﾊﾟﾀｰﾝ･ﾃﾞｰﾀ･ｱﾄﾞﾚｽ
C800           23       ;   EXX
C800           24       ;   CALL SPX
C800           25
C800           26       ;コール例 3 (消す)
C800           27       ;   LD   BC,1008H   ;X=08H,Y=10H
C800           28       ;   LD   DE,0402H   ;消したい
C800           29                           ;ｷｬﾗｸﾀの大きさ
C800           30                           ;XL=4,YL=2
C800           31       ;   CALL KESU
C800           32
C800           33 SPX:
C800           34 ;---------------------
C800           35       ;汎用 疑似ｽﾌﾟﾗｲﾄ
C800           36 ;---------------------
C800           37       ;X=(0-79),Y=(0-49)
C800           38       ;B=Y C=X
C800           39       ;HL=DATA AREA
C800           40       ;HL'=MASK AREA
C800           41       ;DE=SIZE X,Y
C800           42
C800 7A        43           LD      A,D
C801 32 08 C9  44           LD      (XLEN),A
C804 78        45           LD      A,B
C805 0F        46           RRCA
C806 DA 4F C8  47           JP      C,OTHER ;Yｶﾞﾋｮｳｶﾞｷｽｳ ﾅﾗ..
C809           48
C809 E5        49           PUSH    HL
C80A C5        50           PUSH    BC
C80B 26 00     51           LD      H,0
C80D 68        52           LD      L,B
C80E CB 3D     53           SRL     L       ;Y=Y/2
C810 29        54           ADD     HL,HL   ;*2
C811 29        55           ADD     HL,HL   ;*4
C812 29        56           ADD     HL,HL   ;*8
C813 29        57           ADD     HL,HL   ;*16
C814 44        58           LD      B,H
C815 4D        59           LD      C,L
C816 29        60           ADD     HL,HL   ;*32
C817 29        61           ADD     HL,HL   ;*64
C818 09        62           ADD     HL,BC   ;*80
C819 C1        63           POP     BC
C81A 06 00     64           LD      B,0
C81C 09        65           ADD     HL,BC   ;HL=Y*80+X
C81D 01 00 40  66           LD      BC,4000H
C820 09        67           ADD     HL,BC   ;HL=GRAM ADDR
C821 44        68           LD      B,H
C822 4D        69           LD      C,L
C823 E1        70           POP     HL
C824           71
C824           72 LP00:
C824 C5        73           PUSH    BC
C825           74 LP01:
C825 ED 78     75           IN      A,(C)
C827           76
C827 D9        77           EXX
C828 A6        78           AND     (HL)    ;MASK
C829 23        79           INC     HL
C82A D9        80           EXX
C82B           81
C82B B6        82           OR      (HL)    ;DATA
C82C 23        83           INC     HL
C82D           84
C82D ED 79     85           OUT     (C),A
C82F 78        86           LD      A,B
C830 C6 08     87           ADD     A,8
C832 47        88           LD      B,A
C833 30 F0     89           JR      NC,LP01
C835 C6 40     90           ADD     A,40H
C837 47        91           LD      B,A
C838 03        92           INC     BC
C839 15        93           DEC     D       ;X LEN
C83A 20 E9     94           JR      NZ,LP01
C83C C1        95           POP     BC
C83D E5        96           PUSH    HL
C83E 60        97           LD      H,B
C83F 69        98           LD      L,C
C840 01 50 00  99           LD      BC,80
C843 09       100           ADD     HL,BC
C844 44       101           LD      B,H
C845 4D       102           LD      C,L
C846 E1       103           POP     HL
C847 3A 08 C9 104           LD      A,(XLEN)
C84A 57       105           LD      D,A
C84B 1D       106           DEC     E
C84C 20 D6    107           JR      NZ,LP00
C84E C9       108           RET
C84F          109 OTHER:
C84F E5       110           PUSH    HL
C850 C5       111           PUSH    BC
C851 26 00    112           LD      H,0
C853 68       113           LD      L,B
C854 CB 3D    114           SRL     L       ;Y=Y/2
C856 29       115           ADD     HL,HL   ;*2
C857 29       116           ADD     HL,HL   ;*4
C858 29       117           ADD     HL,HL   ;*8
C859 29       118           ADD     HL,HL   ;*16
C85A 44       119           LD      B,H
C85B 4D       120           LD      C,L
C85C 29       121           ADD     HL,HL   ;*32
C85D 29       122           ADD     HL,HL   ;*64
C85E 09       123           ADD     HL,BC   ;*80
C85F C1       124           POP     BC
C860 06 00    125           LD      B,0
C862 09       126           ADD     HL,BC   ;HL=Y*80+X
C863 01 00 60 127           LD      BC,6000H
C866 09       128           ADD     HL,BC   ;HL=GRAM ADDR
C867 44       129           LD      B,H
C868 4D       130           LD      C,L
C869 E1       131           POP     HL
C86A          132
C86A          133 O_LP00:
C86A C5       134           PUSH    BC
C86B          135 O_LP01:
C86B          136           REPT 3          ;REPTからENDMまでの
C86B          137                           ;命令群を3回くりかえす
C86B          138           IN      A,(C)
C86B          139           EXX
C86B          140           AND     (HL)    ;MASK
C86B          141           INC     HL
C86B          142           EXX
C86B          143           OR      (HL)    ;DATA
C86B          144           INC     HL
C86B          145           OUT     (C),A
C86B          146           LD      A,B
C86B          147           ADD     A,8
C86B          148           LD      B,A     ;1-3ｶｲ
C86B          149           ENDM
C86B          149+                          ;命令群を3回くりかえす
C86B ED 78    149+          IN      A,(C)
C86D D9       149+          EXX
C86E A6       149+          AND     (HL)    ;MASK
C86F 23       149+          INC     HL
C870 D9       149+          EXX
C871 B6       149+          OR      (HL)    ;DATA
C872 23       149+          INC     HL
C873 ED 79    149+          OUT     (C),A
C875 78       149+          LD      A,B
C876 C6 08    149+          ADD     A,8
C878 47       149+          LD      B,A     ;1-3ｶｲ
C879          149+                          ;命令群を3回くりかえす
C879 ED 78    149+          IN      A,(C)
C87B D9       149+          EXX
C87C A6       149+          AND     (HL)    ;MASK
C87D 23       149+          INC     HL
C87E D9       149+          EXX
C87F B6       149+          OR      (HL)    ;DATA
C880 23       149+          INC     HL
C881 ED 79    149+          OUT     (C),A
C883 78       149+          LD      A,B
C884 C6 08    149+          ADD     A,8
C886 47       149+          LD      B,A     ;1-3ｶｲ
C887          149+                          ;命令群を3回くりかえす
C887 ED 78    149+          IN      A,(C)
C889 D9       149+          EXX
C88A A6       149+          AND     (HL)    ;MASK
C88B 23       149+          INC     HL
C88C D9       149+          EXX
C88D B6       149+          OR      (HL)    ;DATA
C88E 23       149+          INC     HL
C88F ED 79    149+          OUT     (C),A
C891 78       149+          LD      A,B
C892 C6 08    149+          ADD     A,8
C894 47       149+          LD      B,A     ;1-3ｶｲ
C895          150
C895 ED 78    151           IN      A,(C)
C897 D9       152           EXX
```

```
C898 A6        153          AND     (HL)     ;MASK
C899 23        154          INC     HL
C89A D9        155          EXX
C89B B6        156          OR      (HL)     ;DATA
C89C 23        157          INC     HL
C89D ED 79     158          OUT     (C).A
C89F E5        159          PUSH    HL
C8A0 60        160          LD      H,B
C8A1 69        161          LD      L,C
C8A2 01 B0 37  162          LD      BC,37B0H
C8A5 B7 ED 42  163          SUB     HL,BC
C8A8 44        164          LD      B,H
C8A9 4D        165          LD      C,L
C8AA E1        166          POP     HL       ;4ｶｲ
C8AB           167
C8AB           168          REPT    3
C8AB           169          IN      A,(C)
C8AB           170          EXX
C8AB           171          AND     (HL)     ;MASK
C8AB           172          INC     HL
C8AB           173          EXX
C8AB           174          OR      (HL)     ;DATA
C8AB           175          INC     HL
C8AB           176          OUT     (C).A
C8AB           177          LD      A,B
C8AB           178          ADD     A,8
C8AB           179          LD      B,A      ;5-7ｶｲ
C8AB           180          ENDM
C8AB ED 78     180+         IN      A,(C)
C8AD D9        180+         EXX
C8AE A6        180+         AND     (HL)     ;MASK
C8AF 23        180+         INC     HL
C8B0 D9        180+         EXX
C8B1 B6        180+         OR      (HL)     ;DATA
C8B2 23        180+         INC     HL
C8B3 ED 79     180+         OUT     (C).A
C8B5 78        180+         LD      A,B
C8B6 C6 08     180+         ADD     A,8
C8B8 47        180+         LD      B,A      ;5-7ｶｲ
C8B9 ED 78     180+         IN      A,(C)
C8BB D9        180+         EXX
C8BC A6        180+         AND     (HL)     ;MASK
C8BD 23        180+         INC     HL
C8BE D9        180+         EXX
C8BF B6        180+         OR      (HL)     ;DATA
C8C0 23        180+         INC     HL
C8C1 ED 79     180+         OUT     (C).A
C8C3 78        180+         LD      A,B
C8C4 C6 08     180+         ADD     A,8
C8C6 47        180+         LD      B,A      ;5-7ｶｲ
C8C7 ED 78     180+         IN      A,(C)
C8C9 D9        180+         EXX
C8CA A6        180+         AND     (HL)     ;MASK
C8CB 23        180+         INC     HL
C8CC D9        180+         EXX
C8CD B6        180+         OR      (HL)     ;DATA
C8CE 23        180+         INC     HL
C8CF ED 79     180+         OUT     (C).A
C8D1 78        180+         LD      A,B
C8D2 C6 08     180+         ADD     A,8
C8D4 47        180+         LD      B,A      ;5-7ｶｲ
C8D5           181
C8D5 ED 78     182          IN      A,(C)
C8D7 D9        183          EXX
C8D8 A6        184          AND     (HL)     ;MASK
C8D9 23        185          INC     HL
C8DA D9        186          EXX
C8DB B6        187          OR      (HL)     ;DATA
C8DC 23        188          INC     HL
C8DD ED 79     189          OUT     (C).A
C8DF E5        190          PUSH    HL
C8E0 60        191          LD      H,B
C8E1 69        192          LD      L,C
C8E2 01 B0 47  193          LD      BC,47B0H
C8E5 09        194          ADD     HL,BC
C8E6 44        195          LD      B,H
C8E7 4D        196          LD      C,L
C8E8 E1        197          POP     HL       ;8ｶｲ
C8E9           198
C8E9 30 80     199          JR      NC,O_LP01
C8EB 78        200          LD      A,B
C8EC C6 40     201          ADD     A,40H
C8EE 47        202          LD      B,A
C8EF 03        203          INC     BC
C8F0 15        204          DEC     D        ;X LEN
C8F1 C2 6B C8  205          JP      NZ,O_LP01
C8F4 C1        206          POP     BC
C8F5 E5        207          PUSH    HL
C8F6 60        208          LD      H,B
C8F7 69        209          LD      L,C
C8F8 01 50 00  210          LD      BC,80
C8FB 09        211          ADD     HL,BC
C8FC 44        212          LD      B,H
C8FD 4D        213          LD      C,L
C8FE E1        214          POP     HL
C8FF 3A 08 C9  215          LD      A,(XLEN)
C902 57        216          LD      D,A
C903 1D        217          DEC     E
C904 C2 6A C8  218          JP      NZ,O_LP00
C907 C9        219          RET
C908 00        220 XLEN:    DB      0
C909           221
C909           222 MAKE_MASK:
C909           223          ; IX=SOURCE
C909           224          ; IY=DESTINATION
C909           225          ; HL=COUNT
C909 11 10 00  226          LD      DE,16
C90C           227 MKMSK:
C90C 06 08     228          LD      B,8
C90E           229 MKMSKLP:
C90E DD 7E 00  230          LD      A,(IX+0)
C911 DD B6 08  231          OR      (IX+8)
C914 DD B6 10  232          OR      (IX+16)
C917 2F        233          CPL
C918 FD 77 00  234          LD      (IY+0),A
C91B FD 77 08  235          LD      (IY+8),A
C91E FD 77 10  236          LD      (IY+16),A
C921 DD 23     237          INC     IX
C923 FD 23     238          INC     IY
C925 2B        239          DEC     HL
C926 10 E6     240          DJNZ    MKMSKLP
C928 DD 19     241          ADD     IX,DE
C92A FD 19     242          ADD     IY,DE
C92C 7C        243          LD      A,H
C92D B5        244          OR      L
C92E 20 DC     245          JR      NZ,MKMSK
C930 C9        246          RET
C931           247
C931           248 KESU:
C931           249   ;X=(0-79),Y=(0-49)
C931           250   ;B=Y C=X
C931           251   ;DE=SIZE X,Y
C931           252
C931 F3        253          DI
C932 CD DE C9  254          CALL    DOJI
C935           255
C935 7A        256          LD      A,D
C936 32 08 C9  257          LD      (XLEN),A
C939 78        258          LD      A,B
C93A 0F        259          RRCA
C93B DA 7C C9  260          JP      C,K_OTHER ;Yｻﾞﾋｮｳｶﾞｷｽｳ ﾅﾗ..
C93E           261
C93E C5        262          PUSH    BC
C93F 26 00     263          LD      H,0
C941 68        264          LD      L,B
C942 CB 3D     265          SRL     L        ;Y=Y/2
C944 29        266          ADD     HL,HL    ;*2
C945 29        267          ADD     HL,HL    ;*4
C946 29        268          ADD     HL,HL    ;*8
C947 29        269          ADD     HL,HL    ;*16
C948 44        270          LD      B,H
C949 4D        271          LD      C,L
C94A 29        272          ADD     HL,HL    ;*32
C94B 29        273          ADD     HL,HL    ;*64
C94C 09        274          ADD     HL,BC    ;*80
C94D C1        275          POP     BC
C94E 06 00     276          LD      B,0
C950 09        277          ADD     HL,BC    ;HL=Y*80+X
C951 44        278          LD      B,H
C952 4D        279          LD      C,L
C953           280
C953           281 K_LP00:
C953 C5        282          PUSH    BC
C954 2E 00     283          LD      L,00H
C956           284 K_LP01:
C956 26 08     285          LD      H,8
C958           286 K_LP02:
C958 ED 69     287          OUT     (C),L
C95A 78        288          LD      A,B
C95B C6 08     289          ADD     A,8
C95D 47        290          LD      B,A
C95E 25        291          DEC     H
C95F 20 F7     292          JR      NZ,K_LP02
C961 D6 40     293          SUB     40H
C963 47        294          LD      B,A
C964 03        295          INC     BC
C965 15        296          DEC     D        ;X LEN
C966 20 EE     297          JR      NZ,K_LP01
C968 C1        298          POP     BC
C969 60        299          LD      H,B
C96A 69        300          LD      L,C
C96B 01 50 00  301          LD      BC,80
C96E 09        302          ADD     HL,BC
C96F 44        303          LD      B,H
C970 4D        304          LD      C,L
C971 3A 08 C9  305          LD      A,(XLEN)
C974 57        306          LD      D,A
C975 1D        307          DEC     E
C976 20 DB     308          JR      NZ,K_LP00
C978 ED 78     309          IN      A,(C)    ;ﾄﾞｳｼﾞｱｸｾｽｶｲｼﾞｮ
C97A FB        310          EI
C97B C9        311          RET
C97C           312 K_OTHER:
C97C C5        313          PUSH    BC
C97D 26 00     314          LD      H,0
C97F 68        315          LD      L,B
C980 CB 3D     316          SRL     L        ;Y=Y/2
C982 29        317          ADD     HL,HL    ;*2
C983 29        318          ADD     HL,HL    ;*4
C984 29        319          ADD     HL,HL    ;*8
C985 29        320          ADD     HL,HL    ;*16
C986 44        321          LD      B,H
C987 4D        322          LD      C,L
C988 29        323          ADD     HL,HL    ;*32
C989 29        324          ADD     HL,HL    ;*64
C98A 09        325          ADD     HL,BC    ;*80
C98B C1        326          POP     BC
C98C 06 00     327          LD      B,0
C98E 09        328          ADD     HL,BC    ;HL=Y*80+X
C98F 01 00 20  329          LD      BC,2000H
C992 09        330          ADD     HL,BC    ;HL=GRAM ADDR
C993 44        331          LD      B,H
C994 4D        332          LD      C,L
C995           333
C995           334 K_O_LP00:
C995 C5        335          PUSH    BC
C996           336 K_O_LP01:
C996 21 00 03  337          LD      HL,0300H
C999           338 K_O_LP02:
C999 ED 69     339          OUT     (C),L
C99B 78        340          LD      A,B
C99C C6 08     341          ADD     A,8
C99E 47        342          LD      B,A      ;1-3ｶｲ
C99F 25        343          DEC     H
C9A0 20 F7     344          JR      NZ,K_O_LP02
C9A2           345
C9A2 ED 69     346          OUT     (C),L
C9A4 60        347          LD      H,B
C9A5 69        348          LD      L,C
C9A6 01 B0 37  349          LD      BC,37B0H
C9A9 B7 ED 42  350          SUB     HL,BC
C9AC 44        351          LD      B,H
C9AD 4D        352          LD      C,L      ;4ｶｲ
C9AE           353
C9AE 21 00 03  354          LD      HL,0300H
C9B1           355 K_O_LP03:
C9B1 ED 69     356          OUT     (C),L
C9B3 78        357          LD      A,B
C9B4 C6 08     358          ADD     A,8
C9B6 47        359          LD      B,A      ;5-7ｶｲ
C9B7 25        360          DEC     H
C9B8 20 F7     361          JR      NZ,K_O_LP03
C9BA           362
C9BA ED 69     363          OUT     (C),L
C9BC 60        364          LD      H,B
C9BD 69        365          LD      L,C
C9BE 01 B0 07  366          LD      BC,7B0H
C9C1 09        367          ADD     HL,BC
C9C2 44        368          LD      B,H
C9C3 4D        369          LD      C,L      ;8ｶｲ
C9C4           370
C9C4 03        371          INC     BC
C9C5 15        372          DEC     D        ;X LEN
C9C6 C2 96 C9  373          JP      NZ,K_O_LP01
C9C9 C1        374          POP     BC
C9CA 60        375          LD      H,B
C9CB 69        376          LD      L,C
C9CC 01 50 00  377          LD      BC,80
C9CF 09        378          ADD     HL,BC
C9D0 44        379          LD      B,H
C9D1 4D        380          LD      C,L
C9D2 3A 08 C9  381          LD      A,(XLEN)
C9D5 57        382          LD      D,A
C9D6 1D        383          DEC     E
C9D7 C2 95 C9  384          JP      NZ,K_O_LP00
C9DA ED 78     385          IN      A,(C)
C9DC FB        386          EI
C9DD C9        387          RET
C9DE           388
C9DE           389 DOJI:    ;ﾄﾞｳｼﾞｱｸｾｽﾓｰﾄﾞ SET
C9DE C5        390          PUSH    BC
C9DF 01 02 1A  391          LD      BC,1A02H
C9E2 ED 78     392          IN      A,(C)
C9E4 F6 20     393          OR      32
C9E6 ED 79     394          OUT     (C),A
C9E8 E6 DF     395          AND     0DFH
C9EA ED 79     396          OUT     (C),A
C9EC C1        397          POP     BC
C9ED C9        398          RET
C9EE           399
```

★(で)のショートプロぱーてぃ その10

ちょっとへびぃ…!?

Komura Satoshi 古村　聡

今宵あなたのMZ-1500が甦える。放物線シミュレーションゲーム「GREEN」の登場です。もう1本は，少しリストが長めのX68000用ゲーム「いもむしの……」，いや失礼「へびのぼーけん」です。がんばって入力してみてください。

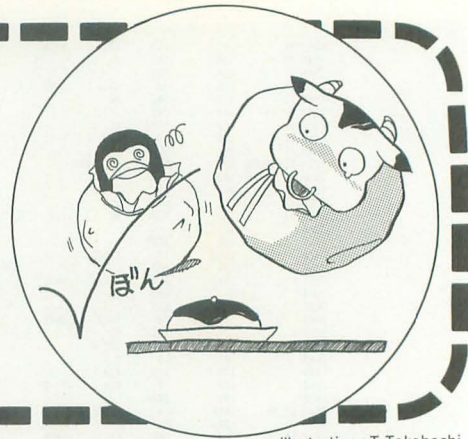



illustration : T.Takahashi

はろー。徹夜明けでいつもの3倍ハイになっている（で）です。えーと，今月号のOh!Xには付録のフロッピーディスクが付いてるんですけど，なんと我がショートプロぱーてぃから先々月号掲載になったsp_chk( )がこのフロッピーに入ってるんです。はい，拍手拍手！　アセンブラとリンカがなくてsp_chk( )が打ち込めなかった方もこれでもう安心。

フロッピーの中にはsp_chk( )のソースリスト（つまり先々月号に載っていたみたいにリストになっているやつね）ではなく実行形式のファイルが入っているのでアセンブラもリンカも必要ないっつーわけです。もちろんディスクはプログラムが圧縮されてるから解凍する必要があるけど。そのへんのことは特集記事を参照のことね。

プログラムの解凍をしたあと，BASICを使っているディスクのBASIC 2のディレクトリにコピーしてBASIC.CNFというファイルにEDで，

FUNC　=　SPR

という1行を加えるだけでOKです。ま，詳しくは特集のほうを見てくださいね。

今回ディスクに入っているsp_chk( )は，ぱーてぃハンズのほうでもスプライトの当たり判定に使っていますので，ハンズのプログラムを打ち込む人には絶対に必要になるプログラムです。またX-BASICでスプライトの判定をするには非常に便利なプログラムですので自作に，投稿にぜひどうぞ（なんか，お茶の間ショッピングみたいだなー。それではお値段です，なんて……)。

楽しいシミュレーション入門

さて，今月の1本目。今月はめずらしくMZ用のゲームが届きました。愛知県の永田さんの投稿でMZ-1500用「GREEN」です。

GREEN

へびのぼーけん

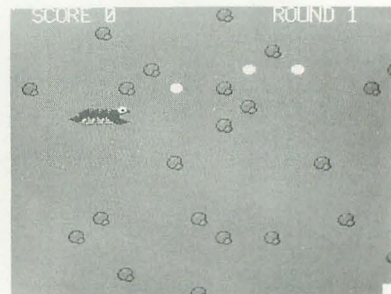
GREEN for MZ-1500

（MZ-5Z001）

愛知県　永田秀史

画面左下に45度，90度の印のついているところから角度とパワーを調整して放物線状に球を飛ばしてうまく右側の赤いグリーン（なんか変な日本語だな）に球をのせてください。球は地面と画面の左右の端で跳ね返りますのでちゃーんと跳ね返りも計算にいれてね（1回も跳ね返らないと点が低いのだよ)。

うーん，わいいこーるにぶんのいちじーてぃーじじょうだ。実は跳ね返り係数まできっちり計算してますからねー，跳ね返り係数が1以上になると地獄ですよー，なんてたって地面に落ちたほうがもとより高く飛びますからねー，球が。

ショートのネタとしては放物線ものっていうのはわりとありがちなパターンなんですが，やっぱりなにかこう，「コンピュータによるシミュレーション入門講座・その1」みたいな雰囲気で人気があるんでしょうね。私はなんとなく物理の授業を思い出しちゃって，ああいうのを作るのは苦手なんですけど。そういえば，うちの学校でも工学系の人たちは慣れないキーボードに苦労してぽちぽち打ち込んで放物線やらサインカーブやら苦労して表示してたなぁ，ごくろうさま。あの人たちってやっぱり研究も物理関係なのかなあ，たいへんだなあ。ああ，私は情報数理科でよかった。なんてたって研究課題が「戦略ゲームを1年で1本作れ」だもんね（まるでどっかのソフトハウスのようだ)。

それにしても，MZ-1500の投稿というのもなかなかめずらしいですねー。QDがもう近所で売っていないとのことで，カセットテープで投稿してくれました。

がんばれX-BASIC!

続いて今月の2本目。今月の2本目は神奈川県の石井さんでBG機能を使ったX68000用横スクロールゲームで「へびのぼーけん」です。

へびのぼーけん　for X68000

（X-BASIC・ジョイスティック専用）

神奈川県　石井幸雄

中国四千年の謎の秘術(なんだそりゃ？)で蛇になった君は飛んでくる岩を避けながら白い秘薬を食べなければなりません。

ルール自体は岩を避けながら餌を食べるだけなので簡単なのですが，なにせ自分が蛇だもんだから身体が長くて岩を避けるのがもう大変。ゲームとしてはよくある横スクロールゲームなんだけど，自分を長くすることで楽しめるようにしてるし，そのうえ音楽はついてるわで（のどかな音楽なんだこれが)，なかなかよく出来たゲームなんです。X-BASICのゲームにしては速いですしね。どの作業をどれにやらせるかっていう命令とか機能とかの見極めがとてもう

まいと思います。あとはもう少しREM文をつけるとかして見やすいプログラムを心がけて作れば完璧でしょう，うん。

ところでX-BASICのゲームを作るとどうしてもネックになってしまうのが，スピード。このゲームはちゃんと遊べるぐらいのスピードになっていますけど，それでもX1なんかで作ってもそんなに変わらないんじゃないかって感じ。X-BASICではゲーム作るときに，うまく作業量をどのぐらいにするかという見極めがうまくいかないとどうしても，「重いよ，遅いよ，困ったよー」という事態になってしまうのですよね。

で，速くするためにいろいろ姑息なことをやったりBASICコンパイラやCに逃げたりとか，先々月のsp_chk()みたいにアセンブラで外部関数を作っちゃうとかになっちゃうんですよね。でも，アセンブラやCはそれ専用の知識が必要だし，BASICコンパイラはインタプリタからコンパイラへ行ったりきたりで面倒臭いわ時間はかかるわでねー。それにどうせBASICコンパイラでコンパイルするプログラムはどのみちX-BASICでデバッグするんだから，インタプリタですいすい速く動いてくれるんならそれに越したことはないんです。関数やらローカル変数が使えて面倒臭いポインタを使わなくてもすむわで言語の仕様自体はとてもよくできてると思うんだけど。

それじゃ，今月はこんなところで。おっと，来月はこのコーナーも1周年になるんですねー，早いもんだ。なんかお祭り企画でもできるといいんですけど，ショートプロぱーてぃ1周年記念ぱーてぃとか（おおっ，リカーシブしてる）。そんなこんなでまた来月。

リスト1 GREEN

```
10 INIT "CRT:G":CLR:PRTY 1
20 NA$="MZ-1500":BS=50:MK=1:TR=9
30 HN=1:SC=0:MS=0
40 C=7:N=.5:AN=N:XN=N:BN=N:DIM A$(3)
50 T=0:TY=0:BB=3:RB=0:TB=0:HI=189
60 CLS 3:RESTORE 990
70 FOR I=0 TO 3:READ A$(I):NEXT
80 CURSOR 0,10:PRINT "GREEN ON ONE!?";
90 CURSOR 0,21:PRINT [4,0]" 90 45↓←←←←←←  |/↓←←←←  V 0↓←←←←←
";
100 HX=INT(RND(1)*30+5):HY=INT(RND(1)*18+6)
110 PU=0:IF HX<15 THEN PU=22
120 CURSOR HX,HY:PRINT [2,0]"______↓←←←←←";:HX=HX*8:HY=HY*8
130 KW=INT(RND(1)*17*HN)
140 IF KW>80 THEN KW=80
150 CURSOR PU,6:PRINT "ｶｸﾄﾞ   ";KW;A$(2)
160 E=RND(1)+.3:E=INT(1000*E):E=E/1000
170 KB=INT(RND(1)*10)
180 IF KB=0 THEN KB=0:BH=0:GOTO230
190 IF KB=>TR THEN KB=TR:BH=3:GOTO230
200 IF KB>=3 AND E<.8 THEN BH=3:GOTO230
210 IF KB<4 AND E>1 THEN BH=2:GOTO230
220 BH=INT(RND(1)*2+2)
230 CURSOR PU,7:PRINT A$(1);KB;A$(BH)
240 CURSOR  0,1:PRINT USING "BEST #####  HOLE ###  &          &";B
S;MK;NA$
250 CURSOR  0,2:PRINT USING "SCORE#####  HOLE ###  ﾊﾈｶｴﾘ ##.##";
SC;HN;E
260 BOX[4,0]HX,HY+8,HX+39,HY+15
270 BOX[4,0]0,-1,319,199
280 BOX[4,0]0,191,31,199
290 CURSOR PU,8:PRINT "ｱﾄ     ";3-MS;" ｶｲ";
300 CURSOR  0,3:INPUT "ｶｸﾄﾞ ";W
310 IF W<KW OR W>180 THEN 290
320 CURSOR  0,4:INPUT "ｼｮｿｸ ";V
330 IF V<=0 OR V>200 THEN 290
340 REM ------------------------------
350 O=π/180*W:VY=V*SIN(O):GV=V*COS(O)
360 X=16+GV*T:Y=HI-(VY*TY-4.9*TY*TY)
370 SET [C,0] X,Y:HI=198
380 GET A$:IF A$=" " THEN RB=TR-1
390 REM ------------------------------
400 VZ=VY-9.8*TY:IF VZ>0 THEN 520
410 IX=INT(X):IY=INT(Y)
420 IF IX<HX-8 OR IX>HX+48 THEN 520
430 IF IY<HY-70 OR IY>HY+12 THEN 520
440 AN=XN/2:BN=N/2:VZ=ABS(VZ):GX=ABS(GV)
450 IF VZ>30 OR GX>30 THEN AN=AN/2:BN=BN/2
460 IF VZ>60 OR GX>60 THEN AN=AN/2:BN=BN/2
470 IF IX<HX OR IX>HX+39 THEN 540
480 IF IY<HY+4 OR IY>HY+12 THEN 540
490 FG=198-INT(VY*VY/19.6)
500 IF FG>HY+10 THEN 540
510 GOTO 590
520 AN=XN:BN=N
530 REM ------------------------------
540 T=T+AN:TY=TY+BN
550 IF X<0 OR X>320 THEN PG=1:GOTO 790
560 IF Y>200 THEN PG=2:GOTO 790
570 GOTO 360
580 REM ﾉｯﾀ --------------------------
590 IF RB>0 AND BH=0 THEN 730
600 IF RB<KB AND BH=2 THEN 730
610 IF RB>KB AND BH=3 THEN 730
620 CURSOR 0,10:PRINT "ｳﾏｸ ﾉｯﾀ!!!    ";
630 IF RB=0 THEN 660
640 CURSOR 0,12:PRINT USING"(ﾀﾒｼ ##)*(ﾊﾈｶｴﾘ ##)*(ｶﾍﾞ ##+3)=### "
;BB;RB;TB;BB*RB*(3+TB)
650 SC=SC+BB*RB*(3+TB):GOTO 680
660 CURSOR 0,12:PRINT USING"(ﾀﾒｼ ##)*(ﾁｮｸｾﾂ 3)=### ";BB;BB*3
670 SC=SC+BB*3
680 FORI=0TO30:SOUND20+I,1:NEXT
690 FORI=0TO2000:NEXT
700 HN=HN+1:GOTO 40
710 REM ﾐｽｯﾀ -------------------------
720 MS=MS+1:IF MS=3 THEN 850
730 CURSOR 0,10:PRINT " ﾊﾞｯﾃﾝ !!      ";
740 PRINT USING" #### ";HN*(-5)
750 SC=SC-HN*5:HN=HN+1
760 FORI=0TO20:BEEP:FORY=0TO5*I:NEXT:NEXT
770 FORI=0TO250:NEXT:GOTO 40
780 REM ﾊﾈｶｴﾘ ------------------------
790 RB=RB+1
800 IF RB=TR THEN 930
810 IF PG=2 THEN VY=VY*E:TY=0:Y=198
820 IF PG=1 THEN XN=-XN:AN=XN:T=T+AN*2:TB=TB+1
830 SET [C,0] X,Y:BEEP:PG=0:GOTO 360
840 REM GAME OVER --------------------
850 CURSOR 0,10:PRINT "ｹﾞｰﾑ ｵﾜﾘ       ";
860 FOR I=0TO23:SOUND50+I,1:FORY=0TO3:SOUND 70-Y*I,1:NEXT:NEXT
870 IF BS>SC THEN 900
880 CURSOR 0,10:PRINT "ｱﾅﾀ ｲﾁﾊﾞﾝ ｲｲﾋﾄ !!"
890 CURSOR 0,11:INPUT "ﾅﾏｴ ｲｯｺ ｲﾚﾙ(10ｼﾞﾏﾃﾞ)";NA$:BS=SC:MK=HN
900 GET A$:IF A$<>"" THEN 30
910 GOTO 900
920 REM ｼｯﾊﾟｲ ------------------------
930 IF C=<6 THEN 720
940 CURSOR 0,10:PRINT"ﾓｳ ｲｯﾁｮ !!      ";
950 FOR I=0 TO 30:SOUND 70-I,1:SOUND 30+I,1:BEEP:NEXT
960 C=C-1:T=0:TY=0:RB=0:XN=N:AN=N:BN=N:BB=BB-1:TB=0
970 IF BB=<0 THEN BB=1
980 GOTO 290
990 DATA " ﾁｮｸｾﾂ ","ﾊﾞｳﾝﾄﾞ"
1000 DATA " ｲｼﾞｮｳ "," ｲｶ "
```

リスト2 へびのぼーけん

```
10 /**********************************************/
20 /*              へびのぼーけん                 */
30 /*              X-BASIC X68000                 */
40 /**********************************************/
50 screen 0,2,1,1
60 sp_init() : sp_clr():m_init()
70 window(0,0,511,511)
80 dim char iwa(255),esa(255),blocks(31,15)
90 dim int hebi(4,1)
100 dim int hoko(9,1)={0,0,-10,8,0,8,6,8,-10,0,0,0
110                   ,6,0,-10,-8,0,-8,6,-8 }
120 int ix,iy,score,gx,stcount,stlong=80,hi
130 int round=1,js,kac1=28000,kac2=25000,bkac,kac
140 str bgm1[99],bgm2[99],bgm3[99],bgm4[99],bgm5[99]
150 str koka1[99],koka2[99]
160 bgm1="@3V10Q5O5L8[DO]G4GGFEF4FFEDE4EEDCDR>GG4&G[LOOP]"
170 bgm2="@3V13O4L8[DO]G4GGFEF4FFEDE4EEDCDR>EE4&E[LOOP]"
180 bgm3="@35V9O4L8[DO]E4EEDCD4DDC>B<C4CC>BABRCC4&C[LOOP]"
190 bgm4="@16V10O3[DO]GGGFFFEEED8R8V5>E8E&E8V10[LOOP]"
200 bgm5="T120@45V12P3O3L8[DO]|:16 A :|AAARAA4&A[LOOP]"
210 koka1="@2 V15 @L1 |:3ABCDEFG :|"
220 koka2="@57 V15 L16 G8GGFEC4"
230 /*
240 for i=1 to 8 : m_alloc(i,2000) : next
250 for i=1 to 8 : m_assign(i,i) : next
260 m_trk(1,"P1"+bgm1) : m_trk(2,"P2 Y49,30"+bgm1)
270 m_trk(3,bgm2) : m_trk(4,bgm3)
280 m_trk(5,bgm4) : m_trk(6,bgm5)
290 m_trk(7,koka1) : m_trk(8,koka2)
300 setpt() : setpal() : apage(1)
310 fill(0,0,511,511,21) : kac=kac1
320 open()
330 end
340 /*
350 func main()
360 cls : apage(0)
370 m_play(1,2,3,4,5,6)
380 for i=0 to 4
390   hebi(i,0)=128-i*8
400   hebi(i,1)=120
410 next
420 /*
430 locate 2,0,0 : print "SCORE";score
440 locate 22,0,0 : print "ROUND";round
450 while 1
460   js=stick(1)
470   move_hebi(js)
480   move_block()
490   disp_hebi()
500   at_chk()
510 endwhile
520 endfunc
```

```
530 /*
540 func move_hebi(ai;char)
550 if ai=0 then for i=0 to 100 : next : return()
560 for i=0 to 3
570   hebi(4-i,0)=hebi(3-i,0)
580   hebi(4-i,1)=hebi(3-i,1)
590 next
600 hebi(0,0)=hebi(0,0)+hoko(ai,0)
610 hebi(0,1)=hebi(0,1)+hoko(ai,1)
620 if hebi(0,0)<0 then hebi(0,0)=0
630 if hebi(0,1)<0 then hebi(0,1)=0
640 if hebi(0,0)>239 then hebi(0,0)=239
650 if hebi(0,1)>239 then hebi(0,1)=239
660 endfunc
670 /*
680 func move_block()
690 int x1,x2,y1,y2
700 for i=0 to 3
710   iy=iy+32
720   if iy=256 then {
730     iy=0 : ix=ix+16
740     if gx=0 then gx=1 else gx=0
750     stcount=stcount+1
760     if stcount=stlong-18 then bkac=kac : kac=32768
770     if stcount=stlong then stclear()
780   }
790   if ix=512 then ix=0
800   if rand()>kac then {
810     x1=(ix+288) mod 512 : y1=iy+gx*16
820     x2=(ix+303) mod 512 : y2=iy+gx*16+15
830     if rand()>kac2 then {
840       put(x1,y1,x2,y2,esa)
850       blocks((ix¥16+18) mod 32,(iy¥16)+gx)=2
860     } else {
870       put(x1,y1,x2,y2,iwa)
880       blocks((ix¥16+18) mod 32,(iy¥16)+gx)=1 } }
890     if blocks(ix¥16,(iy¥16)+gx)<>0 then {
900       fill(ix,iy,ix+15,iy+15+gx*16,0)
910       blocks(ix¥16,(iy¥16)+gx)=0 }
920 next
930 home(0,ix+(iy¥16),0)
940 endfunc
950 /*
960 func disp_hebi()
970 sp_move(1,hebi(0,0),hebi(0,1),1)
980 sp_move(2,hebi(1,0),hebi(1,1),2)
990 sp_move(3,hebi(2,0),hebi(2,1),2)
1000 sp_move(4,hebi(3,0),hebi(3,1),2)
1010 sp_move(5,hebi(4,0),hebi(4,1),3)
1020 sp_disp(1)
1030 endfunc
1040 /*
1050 func at_chk()
1060 int sh,ex,ey
1070 for i=0 to 4
1080 ex=((ix+(iy¥16)+hebi(i,0)+8)¥16) mod 32
1090 ey=((hebi(i,1)+8)¥16) mod 32
1100 sh=blocks(ex,ey)
1110   if sh=1 then over()
1120   if sh=2 and i=0 then eat(ex,ey)
1130 next
1140 endfunc
1150 /*
1160 func over()
1170 m_stop()
1180 m_play(8)
1190 locate 11,7,0 : print "いてえっ！"
1200 locate 7,9,0 : print "Ｇ Ａ Ｍ Ｅ　Ｏ Ｖ Ｅ Ｒ"
1210 for i=0 to 31
1220   for j=0 to 15
1230     blocks(i,j)=0
1240   next
1250 next
1260 cls : stcount=0 : kac=kac1
1270 if score>hi then hi=score
1280 score=0 : round=1
1290 ix=0 : iy=0 : gx=0 : open()
1300 endfunc
1310 /*
1320 func eat(tx,ty)
1330 score=score+100
1340 locate 7,0,0 : print score
1350 blocks(tx,ty)=0 : m_play(7)
1360 fill(tx*16,ty*16,tx*16+15,ty*16+15,0)
1370 endfunc
1380 /*
1390 func stclear()
1400 m_stop()
1410 m_play(8)
1420 locate 4,8,0 : print "Ｒ Ｏ Ｕ Ｎ Ｄ　Ｃ Ｌ Ｅ Ａ Ｒ ！"
1430 round=round+1
1440 for i=0 to 15000 : next
1450 kac=bkac-1300:stcount=0
1460 ix=0 : iy=0 : gx=0
1470 main()
1480 endfunc
1490 /*
1500 func open()
1510 str mes1="へびのぼーけん",mes2="PUSH TRIGGER BUTTON"
1520 str mes3="HI SCORE "
1530 sp_disp(0)
1540 mes3=mes3+itoa(hi)
1550 apage(0) : wipe()
1560 symbol(46,296,mes1,1,4,2,1,0)
1570 symbol(42,292,mes1,1,4,2,56,0)
1580 /*
1590 for i=0 to 255 : home(0,0,i) : next
1600 /*
1610 symbol(80,452,mes3,1,1,1,1,0)
1620 symbol(76,448,mes3,1,1,1,129,0)
1630 while 1
1640   symbol(58,416,mes2,1,1,1,1,0)
1650   symbol(54,412,mes2,1,1,1,129,0)
1660   for i=0 to 1000
1670     if strig(1)<>0 then wipe() : main()
1680   next
1690   fill(54,412,208,436,0)
1700   for i=0 to 5000 : next
1710 endwhile
1720 endfunc
1730 /*
1740 func setpt()
1750 dim char ptn(256)={ 0
1760               , 0, 0, 0, 0, 0, 0, 0, 0,15,15,15, 0, 0, 0, 0
1770           , 0, 0, 0, 0, 0, 0, 0, 0,15,15,15,15,15, 0, 0, 0
1780           , 0, 0, 0, 0, 0,10,10,10,15,15, 1, 1,15,10,10, 0
1790           , 0, 0, 0, 0,10,10,10,10,15,15, 1, 1,15,10,10,10
1800           , 0, 0,10,10,10,10,10,10,10,15,15,15,10,10,10,10
1810           , 0, 0,10,10,10,10,10,10,10,10,10,10,10, 0, 0, 0
1820           , 0,10,10,10,10,10,10,10,10,10,10, 0, 0, 0, 0, 0
1830           , 0,10,10,10,10,10,10,10,10, 0, 0, 0, 0, 0, 0, 0
1840           , 0,10,10,10,10,10,10,10, 0, 0, 0, 0, 0, 0, 0, 0
1850           , 0,10,10,10,10,10,10,10,10,10, 0, 0, 0, 0, 0, 0
1860           , 0,10,10,10,10,10,10,10,10,10,10,10, 0, 0, 0, 0
1870           , 0, 0,10,10,10,10,10,10,10,10,10,10,10,10, 0, 0
1880           , 0, 0, 0,10,10,10,10,10,10,10,10,10,10,10,10,10
1890           , 0, 0, 0, 0, 0,10,10,10,10,10,10,10,10,10, 0, 0
1900           , 0, 0, 0, 0, 0, 0, 0,10,10,10,10, 0, 0, 0, 0, 0
1910           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
1920 sp_def(1, ptn)
1930 /*
1940 ptn={        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
1950           , 0, 0, 0, 0, 0, 0,10,10,10,10, 0, 0, 0, 0, 0, 0
1960           , 0, 0, 0, 0,10,10,10,14,14,10,10,10, 0, 0, 0, 0
1970           , 0, 0, 0,10,10,10,10,10,10,10,10,10,10, 0, 0, 0
1980           , 0, 0,10,10,10,10,10,13,13,10,10,10,10,10, 0, 0
1990           , 0, 0,10,10,10,10,10,10,10,10,10,10,10,10, 0, 0
2000           , 0,10,10,10,10,10,10,12,12,10,10,10,10,10,10, 0
2010           , 0,10,10,10,10,10,10,10,10,10,10,10,10,10,10, 0
2020           , 0,10,10,10,10,10,10,10,10,10,10,10,10,10,10, 0
2030           , 0,10,10,10,10, 3, 6, 6, 6, 6, 3,10,10,10,10, 0
2040           , 0, 0,10, 6, 6, 3, 4, 4, 4, 4, 3, 6, 6,10, 0, 0
2050           , 0, 0, 6, 6, 6, 3, 6, 6, 6, 6, 3, 6, 6, 6, 0, 0
2060           , 0, 0, 0, 6, 3, 3, 4, 4, 4, 4, 3, 3, 6, 0, 0, 0
2070           , 0, 0, 0, 0, 3, 6, 6, 6, 6, 6, 6, 3, 0, 0, 0, 0
2080           , 0, 0, 0, 0, 0, 0, 4, 4, 4, 4, 0, 0, 0, 0, 0, 0
2090           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
2100 sp_def(2, ptn)
2110 /*
2120 ptn={        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2130           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2140           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2150           , 0, 0, 0, 0, 0, 0, 0, 0, 0,10,10,13, 0, 0, 0, 0
2160           , 0, 0, 0, 0, 0, 0, 0,10,14,10,10,13,10,10, 0, 0
2170           , 0, 0, 0, 0, 0, 0,10,10,13,10,10,10,10,10,13, 0
2180           , 0, 0, 0, 0, 0,14,10,10,10,10,10,10,10,10,13, 0
2190           , 0, 0, 0, 0,10,13,10,10,10,10,10,10,10,10,10,10
2200           , 0, 0, 0,10,10,10,10,10,10,10,10,10,10,10,10,10
2210           , 0,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10
2220           ,10,10,10,10,10,10,10,10,10,10,10,10,10, 6, 6, 6
2230           , 0, 0,10,10,10,10,10,10,10,10, 6, 6, 6, 4, 4, 0
2240           , 0, 0, 0, 0,10,10,10,10, 6, 6, 6, 4, 4, 6, 6, 0
2250           , 0, 0, 0, 0, 0, 0, 6, 6, 6, 4, 4, 6, 4, 6, 0, 0
2260           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 6, 4, 6, 0, 0, 0, 0
2270           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
2280 sp_def(3, ptn)
2290 /*
2300 iwa={        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2310           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2320           , 0, 0, 0, 0, 0, 3, 3, 3, 3, 3, 0, 0, 0, 0, 0, 0
2330           , 0, 0, 0, 0, 3, 4, 4, 4, 4, 3, 3, 0, 0, 0, 0, 0
2340           , 0, 0, 0, 3, 3, 5, 5, 5, 4, 4, 4, 3, 0, 0, 0, 0
2350           , 0, 0, 0, 3, 4, 5, 5, 4, 4, 4, 4, 3, 0, 0, 0, 0
2360           , 0, 0, 3, 4, 4, 4, 4, 4, 4, 4, 4, 3, 0, 0, 0, 0
2370           , 0, 0, 3, 4, 4, 4, 4, 4, 4, 4, 4, 3, 0, 0, 0, 0
2380           , 0, 0, 3, 4, 4, 4, 4, 4, 4, 3, 3, 3, 0, 0, 0, 0
2390           , 0, 0, 3, 4, 4, 4, 4, 4, 3, 4, 4, 3, 3, 0, 0, 0
2400           , 0, 0, 0, 3, 4, 4, 4, 4, 3, 5, 5, 4, 3, 0, 0, 0
2410           , 0, 0, 0, 3, 3, 4, 4, 3, 4, 5, 4, 4, 3, 0, 0, 0
2420           , 0, 0, 0, 0, 3, 3, 3, 3, 4, 4, 4, 3, 0, 0, 0, 0
2430           , 0, 0, 0, 0, 0, 0, 0, 0, 3, 3, 3, 0, 0, 0, 0, 0
2440           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2450           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
2460 /*
2470 esa={        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2480           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2490           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2500           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2510           , 0, 0, 0, 0, 0, 0,15,15,15,15, 0, 0, 0, 0, 0, 0
2520           , 0, 0, 0, 0, 0,15,15,15,15,15,15, 0, 0, 0, 0, 0
2530           , 0, 0, 0, 0,15,15,15,15,15,15,15,15, 0, 0, 0, 0
2540           , 0, 0, 0, 0,15,15,15,15,15,15,15,15, 0, 0, 0, 0
2550           , 0, 0, 0, 0,15,15,15,15,15,15,15,15, 0, 0, 0, 0
2560           , 0, 0, 0, 0,15,15,15,15,15,15,15,15, 0, 0, 0, 0
2570           , 0, 0, 0, 0, 0,15,15,15,15,15,15, 0, 0, 0, 0, 0
2580           , 0, 0, 0, 0, 0, 0,15,15,15,15, 0, 0, 0, 0, 0, 0
2590           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2600           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2610           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
2620           , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
2630 endfunc
2640 /*
2650 func setpal()
2660 dim int pal(16)={0,1,1984, 29824
2670                       , 38272, 42496, 48896, 20480
2680                       , 26624, 32768, 38912, 45056
2690                       , 51200, 57344, 63488, 65534}
2700 for i=0 to 15
2710    sp_color(i, pal(i))
2720    palet(i, pal(i))
2730 next
2740 endfunc
```

(で)のぱーてぃハンズ――――(その3)

動け！　我がスプライト

今月は自分の移動にペタペタと手をつけちゃいます。先月と先々月号でお話ししたんですよね，自機や敵やなんかは全部スプライトに描かれているっていうのは（なにそれ一という方はぜひバックナンバーを買ってくださいね，と営業努力もかかさない私であった）。

もうX68000では，アクションゲームとスプライトといえばもう切っても切れない，まるでOh!X編集部と中華料理屋芳珍のような関係ですが，あぁ，このスプライト，思えば遠い道のりであったことよ。スプライト機能があればスプライトに絵を描いておけば，自機だろうが敵だろうが，タマだろうがミケだろうがちょちょいのちょいで動いてくれますが，スプライトがないとそうはいかない。昔のパソコンてえやつはこのスプライトなんかなくてあたりまえ。いや，昔に限らず今もPC-9801なんかにはない。で，そういう機械で1ドットずつ絵を描いたキャラクターを動かそうなんていうともう大騒ぎ。グラフィック画面に書き込んで，消して，書き込んで，消し……。そのうえこのグラフィック画面に書き込むって作業はけっこう時間を食う。あんまり遅いもんだから，な，なんかチラついてない？とか，おーい，遅くてゲームにならんぞぉ，なんていうのは割と日常茶飯事だったりしたんですねー。

で，その対策に，移動しない場所には書き込まないとか某機種だとXORで書き込むだとか姑息な手段ばっかり覚えちゃって，あぁ，思えばむなしい青春だったことよ（おいおい，まだそんな年じゃないって）。X1なんかだとPCGなどという機能もありましたけど，これも8ドット単位にしか動かせませんでしたしね。ほんとにいい時代になったもんです，しみじみ（これでもうちょいX-BASICが速けりゃねー，というのは禁句なのであった）。

で，そのスプライトの動かし方なんですが，そのまえに前回やっていることを順を追って見ていきます（え，遅い？　イライラしないでのんびりいきましょうよ，ね）。

まずスプライトが書き込めてディスプレイに見えるような画面にする，早い話が画面の初期化をします。まず110行の，

110 screen 0, 2, 1, 1

画面は256色使えて画面が256×256ドット，解像度がHighでVRAMをONにします，といっても今回はゲームの場合スプライトが大きく見えるように画面サイズを256×256にしたかっただけなんですけどね。

おおっと，これだけじゃ，いけない。スプライト関係は実はあと少しばかり命令が必要。それが120行です。ま，この行は決まったおまじないみたいなもんです（開け，スプライト！）。

110行にも120行にもなにやら大量に数字が出てきていますが，どの数字がどの機能に対応するかはBASICのマニュアルをよく見てくださいね。関係ない話ですが，「マニュアルっていうのはパソコン語（この場合はBASICですね）の辞書だと思え！」っていうのがプログラムの鉄則だと思います。ほら，よく学校で英語の先生が「とにかく辞書を引け！」ってうるさくいうけど，あれと同じことなんじゃないかなーと私は思うわけです。人のプログラムにわからないところがあったら辞書を引く。なんか書こうと思って辞書を引く。その感覚なんですよね。うーん，辞書なんか大っ嫌いで，なんか思わず教科書の後ろのほうを見ちゃったりとか，えーい投げちゃえー，とかやってた私のセリフじゃないなー，ふぉっふぉっふぉっ。

それからスプライトに書き込む絵を定義します。1240行から1400行までが絵で使う色，つまりパレット。1420行から2960行までが絵の中身ですね。

よしよし，これでお膳立ては揃ったと。さーて，ここでやっとキャラを，自分を最初にいてほしいところにおきます。それが150行です。

150 sp_set (33, x, y, &H121)

このsp_setっていうのはスプライトを好きな場所に置く命令ですよね。このx, yというのが画面上の位置で100行でxに128, yに128が入ってますから，(128, 128) の座標に自分のキャラクターを書いた33番を&h21のプレーンに書いてくれるわけですね。プレーンっていうのは，うーん，ま，優先順位みたいなもんでこれの小さいヤツほど絵が重なったときに上になって表示されると……ま，とりあえず&h1のあとに2ケタつけるものなんです，はい（ちなみに&hのあとの1はパレット番号。sp_color (0, 0, 1) のいちばん右の1になっているパレットを指定しています）。

で，やっと今月のリストの解説

今月のリストはジョイスティックの動きに合わせて自機の位置を変える部分です。160〜340行でメインルーチンをループさせてます。もしゲームオーバーの処理をちゃんとやらせたきゃ340行のあとにそれ用のルーチンをくっつけて，自機がやられたときにflを1以外にすればできるようになっているわけです（今回はショートなんでそこまでやりませんけどねー）。

そしてジョイスティックの動きに合わせて自分の座標を変えます。自分の座標はx, yでしたよね。ジョイスティックは8方向に動きますよね。その値は200行のstick命令でわかるんですが，

		上		
	7	8	9	
左	4	0	6	右
	1	2	3	
		下		

こんな値が返ってくるんです。これって真ん中が0じゃなくて5の場合，左右だけで見ると（つまり左上や左下なんかも左に含めると），

左は3で割ると余り1
中は3で割ると余り2
右は3で割ると余り0

になりますよね。てことは1引いて3で割った余りからさらに1引いたものをX座標に足してやれば変えたいX座標になるわけです。今回は動きを速くするために8ドット単位で動かしたいので途中で8倍しています。その作業をやっているのが230〜250行です。

250行は画面からはみ出さないようにチェックしているところです。見ればわかるようにスプライトの動く範囲はX＝192までです。自分で右端にスコアでも表示できるように改造してほしいなーなどと思っているわけです。そのあとはY座標について同じようなことをやっています。

そして新しい座標に自機を移す。ああ，スプライト。

sp_set (33, x, y, &H121)

これだけでOKなんですからねー。スプライトでなかったら，例の地獄の作業が待っているわけです。うーん，X68000でよかった。さて，ページも予定の枚数をはるかにオーバーしてしまいました。来月はそうですねー，タマを撃って，ついでに敵の移動でもできればいいんですけどねー，どうなるでしょう？　予定は変更するためにある，とあらかじめ言っておこう。では来月またこのOh!Xで。ドドドドド（と車で去る）。

```
160 while (fl=1)
170 /*自機の移動*/
180 /*つーことは、ここがメインなわけだ*/
200 i=stick(1)
210 if i=0 then i=5
220 while (i<>5)
230     j=i-1:j=(j mod 3)
240     x=((j-1)*8)+x
250     if x<16 then x=16 else if x>192 then x=192
260     i=i-1:j=((i¥3)-2)*-1
270     y=((j-1)*8)+y
280     if y<32 then y=32 else if y>256 then y=256
290     sp_set(33,x,y,&H121)
300     i=5
310 endwhile
340 endwhile
```

Oh!X L・I・V・E・in・'90

X1/turbo用 ©Arsys SOFTWARE
ナイトアームズより **3Dステージメインテーマ**
Nishikawa Zenji 西川 善司

X68000用 ©KONAMI
悪魔城伝説より **Beginning**
Tachikawa Masayuki 立川 正之

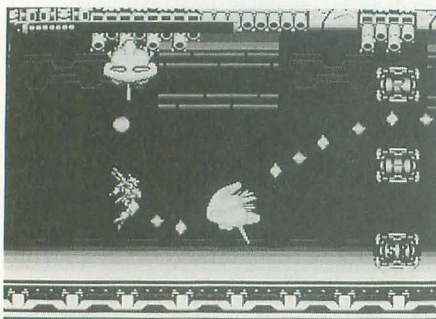
X68000用
この木なんの木
Syouji Shingo 荘司 真吾

今月はゲームミュージック2本とCMソング1本です。ゲームミュージックのほうはアルシスの「ナイトアームズ」とコナミの「悪魔城伝説」の2作品ですが，両方ともFM音源とは思えないほどの出来，読者の方も参考にしてみてください。そしてCMソングは日立の「この木なんの木」。リストも短いことだし，ぜひ打ち込んで聴いてみてください。

まずはナイトアームズ

アルシスソフトの「ナイトアームズ」より，3Dステージメインテーマです。実行には，リスト2の音色セットルーチンが必要ですので，プログラム本体を実行させる前に必ずLOADMしておきます。

さて，プログラムですが，例によってFM8+PSG3=11和音で鳴っています。PSGはバッキングとハイハットで使用しています。ハイハットはPSGのノイズです。Sコマンドにてエンベロープを作り出しています。FM音源をハイハットで1チャンネル潰すのは少しもったいないな，と思ったらこのパターンでいくといいでしょう，参考にしてください。

FMドラムは私が最近よく使う音色です。特に，スネアはFM音源のわりにはアタックが強くて，結構気に入っています。癖がないのでなんにでも使えそうな音色ですから，使えるなと思ったらドンドン使ってみてください。 (善)

ナイトアームズ

FM音源 VS（バーサス） VRC6

コナミからファミコン用に発売されている悪魔城伝説より，1面の曲をお送りしましょう。作者はあのYコマンドの立川君なのです。以前のFM音源講座などを覚えている人もいるかもしれませんね。今回はFM音源講座ではありませんが，リスト中に彼のコメントが載っています。とりあえず，コナミの原曲やテクニックなどの詳しいことは，そちらに譲るとしましょう。なんといっても出たがりの彼のことですから，いいたいことはコメントの中に書いてあることでしょう。ちなみに，曲のスゴさは入力すればわかることです。毎度のことですが，FM音源の音色作りのうまさは天下一品といえます。ぜひ研究してみたいところですね。そのうち音色の作成講座もお願いしたいなぁ(催促)。それから彼のプログラムの特徴として，Yコマンドを多用することが挙げられます。OPMDなどのサンプリングデータを出力することもYコマンドですが，直接OPMをドライブすることもできるのです。MMLで用意されたコマンドだけでは，満足いくまでコントロールできないときなどに使います。あまりにも多いと入力しづらいかもしれませんが，これは彼のこだわりなのかもしれません。

リスト4のコンフィギュレーションファイルで，今月号のおまけDISKに入っているサンプリングデータをOPMDに組み込む必要があります。
(例：A>OPMD /A AKUMA.CNF)

この〜木なんの木，気になる木〜

う〜むコワイ。実にコワイ。小見出しを見て気づいた人も大勢いると思いますが，何を隠そう，この曲は日立の超有名なCMソングです。ああ，目を閉じればあの大きな木が浮かんできます。さらに日立グループのスクロールが……。ちなみに何がコワイって，この曲は今まで約3年間の歴史を誇るこのコーナーにおいて初の投稿だったからなのです。誰でも考えつきそうで誰もがやらなかった，まさにコロンブスの卵のような投稿です。しか〜し，時を隔てること2日，MZ-2000用のこの曲が送られてきているではないですか！ もちろん，別の人からです。友達という可能性もありますが，おそらく違うと思います。残念ながら，MZ用は編集室に届いた時点でディスクがクラッシュしていました。心あたりのある人は申し訳ありませんが，荘司君との関係を明記の上，もう一度投稿してみてください。そうそう，このプログラムの解説としましては一直線にイロモノですので，テクニックよりもアイデアです。ただし，音の選び方や終わらせ方はきちんと考えてありますね。ひょっとしたらキーが違うかもしれませんとは本人の弁ですが，“そんなこともギャグのうち”とでも思えばなんのその。ちなみにキーはあっていると思います。ほとんど1画面プログラムですので，1時間もかからずに打ち込めるでしょう。存分に楽しんでください。 (S.K.)

悪魔城伝説

この木なんの木

```
10 '%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
20 '%                                                     %
30 '%           KNIGHT ARMS 3D STAGE MAIN THEME           %
40 '%                                                     %
50 '%                 (C) ARSYS SOFTWARE                  %
60 '%                                                     %
70 '%              ARRANGED BY Z.NISHIKAWA                %
80 '%                                                     %
90 '%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
100 DEFSTR A-Z:DEFINT I,J,N,V:CLEAR&HFF00:CLS0:TEMPO0
110 DEFUSR=&HB000:DIM V(4,10)
120 DEFFNV$(N,V(0,0))=USR(CHR$(N)+MKI$(VARPTR(V(0,0))))
130 KEY 10,"SAVE"+CHR$(34)+"1:KNIGHT3D"
140 GOTO 190
150 '%%%%% READ VOICE DATA %%%%%
160 FOR J=0 TO 10:FOR I=0TO 4:READ V(I,J):NEXT:NEXT
170 RETURN
180 '%%%%% PROGRAM START %%%%%
190 PLAY "T152";
200 'SOUND NUMBER 1     MELODY
210 '     AF  OM  WF  SY SP  PMD AMD PMS AMS PAN /
220 DATA  50, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
230 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
240 DATA  20,  0,  0, 15,  8, 26,  2,  3,  0,  0,0
250 DATA  25,  0,  0, 15, 15, 21,  2,  1,  0,  0,0
260 DATA  25,  0,  0, 15, 10, 32,  3,  1,  0,  0,0
270 DATA  24,  0,  0, 10,  5,  0,  2,  1,  0,  0,0
280 GOSUB150:A=FNV$(1,V)
290 'SOUND NUMBER 2     BASS
300 '     AF  OM  WF  SY SPD PMD AMD PMS AMS PAN /
310 DATA  50, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
320 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
330 DATA  30, 13,  8, 15, 15, 28,  2,  2,  5,  0,0
340 DATA  30,  8,  5, 15,  5, 30,  3, 10,  2,  0,0
350 DATA  30, 11,  5, 15, 10, 30,  2,  0,  1,  0,0
360 DATA  30,  5,  8, 15,  8,  0,  1,  0,  2,  0,0
370 GOSUB150:A=FNV$(2,V)
380 'SOUND NUMBER 3     BACKING 1
390 '     AF  OM  WF  SY SPD PMD AMD PMS AMS PAN /
400 DATA  59, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
410 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
420 DATA  16,  6,  4, 15, 10, 28,  1,  2,  0,  0,0
430 DATA  25,  6,  2, 15,  5, 30,  0,  2,  0,  0,0
440 DATA  25,  5,  2, 15,  5, 26,  0,  1,  0,  0,0
450 DATA  27,  4,  6, 15,  8,  5,  0,  1,  0,  0,0
460 GOSUB150:A=FNV$(3,V)
470 'SOUND NUMBER 4     BACKING 2
480 '     AF  OM  WF  SY SPD PMD AMD PMS AMS PAN /
490 DATA  59, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
500 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
510 DATA  16,  6,  4, 15, 10, 30,  1,  1,  0,  0,0
520 DATA  25,  6,  2, 15,  5, 30,  0,  1,  0,  0,0
530 DATA  25,  5,  2, 15,  5, 25,  0,  1,  0,  0,0
540 DATA  30,  0,  0, 15,  0,  3,  0,  1,  0,  0,0
550 GOSUB150:A=FNV$(4,V)
560 'SOUND NUMBER 5     BACKING 3
570 '     AF  OM  WF  SY SPD PMD AMD PMS AMS PAN /
580 DATA  33, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
590 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
600 DATA  30, 15,  8, 15,  9, 24,  2,  8,  5,  0,0
610 DATA  30, 13, 21, 15,  5, 24,  3,  4,  0,  0,0
620 DATA  30, 12,  8, 15, 12, 28,  1,  2,  1,  0,0
630 DATA  30,  5,  1, 15,  8,  3,  2,  2,  0,  0,0
640 GOSUB150:A=FNV$(5,V)
650 'SOUND NUMBER 6     BELL
660 '     AF  OM  WF  SY SPD PMD AMD PMS AMS PAN /
670 DATA  51, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
680 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
690 DATA  26, 15,  0, 10,  8, 15,  3, 14,  7,  1,0
700 DATA  26, 15,  0, 12,  6, 10,  3,  4,  0,  2,0
710 DATA  30, 18,  1, 12,  3, 12,  3,  1,  0,  2,0
720 DATA  30,  8, 12, 14,  4,  3,  3,  1,  3,  0,0
730 GOSUB150:A=FNV$(6,V)
740 'SOUND NUMBER 7     SNARE
750 '     AF  OM  WF  SY SPD PMD AMD PMS AMS PAN /
760 DATA  60, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
770 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
780 DATA  31,  9,  8,  9, 10,  0,  0,  0,  4,  0,0
790 DATA  31, 15, 14,  5,  9,  0,  1,  0,  4,  0,0
800 DATA  31, 20, 18,  8, 11,  0,  1,  0,  4,  0,0
810 DATA  31, 10, 10,  9,  9,  0,  1,  0,  4,  0,0
820 GOSUB150:A=FNV$(7,V)
830 'SOUND NUMBER 8     BASS DRUM
840 '     AF  OM  WF  SY SPD PMD AMD PMS AMS PAN /
850 DATA  59, 15,  0,  0,  0,  0,  0,  0,  0,3,  0
860 '     AR  DR  SR  RR  SL  OL  KS  ML DT1 DT2 AM-EN
870 DATA  22,  0,  0, 10,  0, 13,  0, 10,  0,  0,0
880 DATA  26, 26,  0, 10, 15, 19,  0, 13,  0,  3,0
890 DATA  26, 22,  0, 11, 15, 11,  0,  0,  0,  1,0
900 DATA  30, 14,  0,  7, 15,  0,  1,  1,  0,  0,0
910 GOSUB150:A=FNV$(8,V)
920 '
930 '            MELODY
940 '
950 a="L8 e4rf+4g4f+g.a.e2d e4rf+4g4f+g4ra2&a8 e4rf+4g4f+g.a.e2d
 e4rf+4g4f+{gecaf+d}1
960 b1="L16 =0a32&q0a+32=1b4.a2&a& a4.&aq8=0eg&f+&e&f+&d&c&<b&g&
 g+48&a48&a+48&b&>c&<b8>q0b8.q8=1g2& g2&gq8=0>c&<b&a&e&g&f+&d&
970 b2="L16 <b8ra8.&>d8&e&f+&g&f+&e&d&e8 e&c&d&e&f+&d&e4&e&d+48&
d48&c+48&c&c+48&d48&d+48&e48&f48&f+48&q0g =1q8a1 c4r8d2e&q0f
980 c1="L16 =1q8e1=0 >g&e&c&a&f+&d&<b4.b&aa&>q0d q8=1<b4.&=0a24&
g+24&g24&f+&g&a&a+&b&>e&d&<e& >d&c+&c&d&<b&g&a&b8&a&f+&d&<b&a&g&
q0f+
990 c2="L16 =1q8e2&e8&=0f+&g&a&b&>e&<b& a&bb&>dd&gg&a&b&ee&d&g&f
+&d&<b& >g&f+&d&<a&g&f+&d&d+32&e32&g&g+32&a32&a+&b&>d&e&g&>&d&L1
2c&<b&a&g&e&c&a&g&f+&d&<&a&q0f+
1000 d1="L2 =1q8e&_1e&_ e&_e&_ e&_q0e ~5q8>L4e&e&_2e&_L16e&_e&_e
&_e
1010 d2="L16 aa<aa>aa>aa<<aa>aa>aa<aa cccdddeeeeee<c>c<cc> aa<aa
>aa>aa<<aa>aa>aa<a<a32a32 L12aaaaaabbbbbb
1020 PLAY "[  i4  p2 o4 q8 v12 k5  s2,2,0,8 =1 h3"+a;
1030 PLAY "   i1  p3 o4     v13 k5  s2,2,0,12=1 h4"+b1;:PLAY b2;
1040 PLAY c1;:PLAY c2;
1050 PLAY d1;
1060 PLAY "   i5     o4 q8 v10 k5              =0   "+d2+"]";
1070 '
1080 '          ECHO PART
1090 '
1100 aa="L16"+STRING$(2,"c&eg&ec<b>b<b>")+"r8ge caf+d aeL32crdr
erf+rgrf+r L16"+STRING$(2,"c&eg&ec<b>b<b>")
1110 a1=aa+"r8cef&acef&a<b>ef+&a<b>e"
1120 a2=aa+" L24rrrr<a12 aaa12a12 >e12e12<b12 bb>e12e12
1130 d1="L2 =1q8e&_1e&_ e&_e&_ e&_q0e ~5q8>L4e&e&_2e&_e16
1140 d2="L16 rraa<aa>aa>aa<<aa>aa>aa<aa cccdddeeeeee<c>c<cc> aa<
aa>aa>aa<<aa>aa>aa<a<a32a32 L12aaaaaabbbbb24
1150 PLAY ":  i5  p3[o4 q8 v9  k5              =0   "+a1;:PLAY a2;
1160 PLAY "r8.i1  p3 o4     v11 k10 s2,2,0,12=1 h4"+b1;:PLAY b2;
1170 PLAY c1;:PLAY c2;
1180 PLAY d1;
1190 PLAY "   i5     o4 q8 v8  k10             =0   "+d2+"]";
1200 '
1210 '          BACKING
1220 '
1230 a="L8 e4rf+4g4f+g.a.e2d e4rf+4g4f+g4ra2&a8 e4rf+4g4f+g.a.e2
d e4rf+4g4f+{gecaf+d}1
1240 b="L8"+STRING$(2,"e4rf+4g4f+g.a.e2d e4rf+4g4f+g4ra2&a8")
1250 PLAY ":  i4[ p1 o4 q8 v12 k12 s2,2,0,8 =1 h3"+a;
1260 PLAY "       p3        v11 k5             "+b;
1270 PLAY a;
1280 PLAY a+"]";
1290 '
1300 a="L8 g4rb4>c4<b>c.d.<a2g g4rb4>c4<b>c4rd2&d8< g4rb4>c4<b>c
.d.<a2g g4rb4>c4<b>{c<af>d<af+}1
1310 b="L8"+STRING$(2,"g4rb4>c4<b>c.d.<a2g g4rb4>c4<b>c4rd2&d8<"
)
1320 PLAY ":  i3[ p1 o3 q8 v11 k5  s2,2,0,8 =1 h3"+a;
1330 PLAY b;
1340 PLAY a;
1350 PLAY a+"]";
1360 '
1370 a="L8 c4rd4e4de.f+.c2<b> c4rd4e4de4rf+2&f+8 c4rd4e4de.f+.c2
<b> c4rd4e4d{ec<a>f+d<a}1>
1380 b="L8"+STRING$(2,"c4rd4e4de.f+.c2<b> c4rd4e4de4rf+2&f+8")
1390 PLAY ":  i3[ p2 o4 q8 v11 k5  s2,2,0,8 =1 h3"+a;
1400 PLAY b;
1410 PLAY a;
1420 PLAY a+"]";
1430 '
1440 '          BASS
1450 '
1460 aa="L16a_2a~a<aa>aa_<aa>a~a<aa_aa~a> a_aa~g_gg~f_<ff~ff_f>f
f~gg a_a~a<aa>aa_<aa>a~a<aa_aa~a>
1470 a=aa+"f_ff~ff_f~ee_e~ee_e~eeee"+aa+"L12fff fdd24d24 eee eee
1480 b=STRING$(2,aa+"f_ff~ff_f~ee_e~ee_e~eeee")
1490 PLAY ":  i2  p3[o3 q8 v12 k5  s2,2,0,8 =1 h3";:PLAY a;
1500 PLAY b;
1510 PLAY a;
1520 PLAY a+"]";
1530 '
1540 '          BACKING 2
1550 '
1560 aa="L16"+STRING$(2,"g&ec<b>b<b>c&e")+"gecaf+daeL32crdrerf+r
grf+rerdr L16"+STRING$(2,"g&ec<b>b<b>c&e")
1570 a1=aa+"cef&acef&a<b>ef+&a<b>ef+&a
1580 a2=aa+"L24<a12 aaa12a12 >e12e12<b12 bb>e12e12f+12f+12
1590 PLAY ":  i5  p3[o4 q8 v11 k5                      "+a1;:PLAY a2;
1600 PLAY a1;:PLAY a1;
1610 PLAY a1;:PLAY a2;
1620 PLAY a1;:PLAY a2+"]";
1630 '
1640 '          DRUMS
1650 '
1660 aa="L16 i6o5v12g32g32ggdv14o2i7ci8c8.c4i7c4"+STRING$(2,"i8c
4i7c4i8c4i7c4")
1670 a=aa+"i7c4cc8.i8c4i6o5v12e32e32eee"+aa+"L12i8cci7ccc24c24ci
8cci7cccc
1680 b="L16"+STRING$(3,"i8c4i7c4i8c4i7c4")+"i7c4cc8.i8c4i6o5e32e
32eee"+aa+"i7c4cc8.i8c4i6o5v12e32e32eee
1690 PLAY ":        p3    q8     k5                    [";:PLAY a;
1700 PLAY b;
1710 PLAY a;
1720 PLAY a+"]";
1730 '
1740 '          HI-HAT (PSG)
1750 '
1760 a="L16"+STRING$(8,"ff32f32ff ffff32f32 ffff ffff")
1770 PLAY ":  v12 s4,1,12,0 =3 y7,49 y6,8[";:PLAY a;
1780 PLAY a;
1790 PLAY a;
1800 PLAY a+"]";
1810 '
1820 '          PSG BACKING
1830 '
1840 aa="L16"+STRING$(2,"gec<b>b<b>ce")+"gecaf+daeL32crdrerf+rgr
```

```
f+rerdr L16"+STRING$(2,"gec<b>b<b>ce")
1850 a1=aa+"cefacefa<b>ef+a<b>ef+a
1860 a2=aa+"L24<a12 aaa12a12 >e12e12<b12 bb>e12e12f+12f+12
1870 PLAY ":  v12 s0,0,0,0 =1 ^1 k0 [ o4    "+a1;:PLAY a2;
1880 PLAY a1;:PLAY a1;
1890 PLAY a1;:PLAY a2;
1900 PLAY a1;:PLAY a2+"]";
1910 '
```

```
1920 aa="L16"+STRING$(2,"cegec<b>b<b>")+"r8ge caf+d aeL32crdr er
f+rgrf+r L16"+STRING$(2,"cegec<b>b<b>")
1930 a1=aa+"r8cefacefa<b>ef+a<b>e"
1940 a2=aa+" L24rrrr<a12 aaa12a12 >e12e12<b12 bb>e12e12
1950 PLAY ":  v10 s0,0,0,0 =1 ^1 k2 [ o4    "+a1;:PLAY a2;
1960 PLAY a1;:PLAY a1;
1970 PLAY a1;:PLAY a2;
1980 PLAY a1;:PLAY a2+"]"
```

リスト2　音色セットルーチン

```
B000 FE 03 C2 60 20 F5 E5 78 : 95
B008 FE 03 C2 DC B0 1A 13 B7 : 33
B010 CA 6F 20 FE 29 D2 6F 20 : E1
B018 3D 26 00 6F 29 29 44 4D : B5
B020 29 29 29 09 01 90 B1 09 : CF
B028 D5 DD E1 DD 5E 00 DD 56 : 01
B030 01 D5 DD E1 DD 7E 12 0F : 10
B038 0F DD 86 00 77 23 DD 7E : 67
B040 0E 07 07 07 07 DD 86 10 : 9D
B048 77 23 06 06 DD E5 10 FC : 74
B050 11 16 00 06 04 DD 7E 26 : B2
B058 07 07 07 07 DD 86 24 77 : 1A
B060 23 DD 19 10 F0 DD E1 06 : DD
B068 04 DD 7E 20 77 23 DD 19 : 0F
B070 10 F7 DD E1 06 04 DD 7E : 2A
B078 22 0F 0F DD 86 16 77 23 : 53
--------------------------------
SUM: 07 5A A8 78 8D 7A 72 F1 D8B5
```

```
B080 DD 19 10 F2 DD E1 06 04 : C0
B088 DD 7E 2A 0F DD 86 18 77 : 86
B090 23 DD 19 10 F3 DD E1 06 : E0
B098 04 DD 7E 28 0F 0F DD 86 : 08
B0A0 1A 77 23 DD 19 10 F2 DD : 89
B0A8 E1 06 04 DD 7E 1E 07 07 : 72
B0B0 07 07 DD 86 1C 77 23 DD : 04
B0B8 19 10 F0 DD E1 06 05 36 : 18
B0C0 00 23 10 FB DD 7E 08 77 : 08
B0C8 23 DD 7E 0A F6 80 77 23 : 98
B0D0 DD 7E 0C 77 23 DD 7E 04 : 60
B0D8 77 23 36 00 E1 F1 C9 00 : 6B
B0E0 00 00 00 00 00 00 00 00 : 00
B0E8 00 00 00 00 00 00 00 00 : 00
B0F0 00 00 00 00 00 00 00 00 : 00
B0F8 00 00 00 00 00 00 00 00 : 00
--------------------------------
SUM: 73 86 95 D2 27 CA C3 9C C841
```

リスト3　悪魔城伝説

```
  10 /*                                sa."akuma
  20 /*             そんじゃあまあ、仕方ないんで
  30 /*
  40 /*                 「悪魔城伝説」
  50 /*
  60 /*                     より、
  70 /*
  80 /*              ～Begining～
  90 /*
 100 /*             なんでおわしまふ。あびょ。
 110 /*
 120 /*     Programed  by  立川正之  on 1990/4/23
 130 /*
 140 dim char v(4,10)={
 150 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  Bass  1
 160      61,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 170 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 180      31,  23,   0,  15,   4,  10,   0,   2,   0,   0,   0,
 190      23,  16,   4,  15,   4,   2,   0,   1,   0,   0,   0,
 200      23,  16,   4,  15,   4,   2,   0,   2,   0,   0,   0,
 210      23,  16,   4,  15,   4,   2,   0,   4,   0,   0,   0}
 220 m_vset(71,v)
 230 /*
 240 v={
 250 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
 VRC  Bass  2
 260      61,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 270 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 280      31,  23,   0,   4,   4,  10,   4,   2,   4,   4,   4,
 290      23,  16,   4,   4,   4,   2,   4,   1,   4,   4,   4,
 300      23,  16,   4,   4,   4,   2,   4,   2,   4,   4,   4,
 310      23,  16,   4,   4,   4,   2,   4,   4,   4,   4,   4}
 320 m_vset(72,v)
 330 /*
 340 v={
 350 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  1
 360      57,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 370 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 380      31,   0,   0,  15,   0,  22,   0,   2,   0,   0,   0,
 390      31,   0,   0,  15,   0,  32,   0,   4,   0,   0,   0,
 400      31,   0,   0,  15,   0,  32,   0,   4,   0,   0,   0,
 410      31,  16,   6,  15,   2,   0,   0,   2,   0,   0,   0}
 420 m_vset(73,v)
 430 /*
 440 v={
 450 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  Bass  3
 460      61,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 470 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 480      31,  23,   0,   0,   4,  10,   0,   2,   0,   0,   0,
 490      23,  23,   5,  10,   0,   6,   0,   1,   0,   0,   0,
 500      23,  23,   5,  10,   0,   6,   0,   2,   0,   0,   0,
 510      23,  23,   5,  10,   0,   6,   0,   4,   0,   0,   0}
 520 m_vset(74,v)
 530 /*
 540 v={
 550 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
 VRC  Bass  4
 560      61,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 570 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 580      31,  23,   0,   0,   4,   9,   4,   2,   4,   4,   4,
 590      23,  16,   4,   4,   4,   4,   4,   1,   4,   4,   4,
 600      23,  16,   4,   4,   4,   4,   4,   2,   4,   4,   4,
 610      23,  16,   4,   4,   4,   4,   4,   4,   4,   4,   4}
 620 m_vset(75,v)
 630 /*
 640 v={
 650 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
 VRC  2
 660      57,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 670 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 680      31,   0,   0,   0,   0,  23,   0,   4,   0,   0,   0,
 690      31,   0,   0,   0,   0,  23,   0,   2,   0,   0,   0,
 700      31,   0,   0,   0,   0,  33,   0,   4,   0,   0,   0,
 710      20,  15,   6,   7,   3,   0,   0,   2,   0,   0,   0}
 720 m_vset(76,v)
 730 /*
 740 v={
 750 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
Hi-Hat(Close)
 760      44,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 770 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 780      31,   6,   5,  15,   4,   0,   0,  15,   3,   1,   0,
 790      31,  16,  15,  15,   1,  15,   0,   3,   3,   2,   0,
 800      31,  20,   5,  15,   3,   0,   0,   0,   7,   3,   0,
 810      31,  26,  25,  15,   5,  10,   2,   8,   7,   3,   0}
 820 m_vset(77,v)
 830 /*
 840 v={
 850 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
Hi-Hat(Open)
 860      44,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 870 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 880      31,   6,   5,  15,   4,   0,   0,  15,   3,   1,   0,
 890      31,  10,   4,  15,   8,  20,   0,   3,   3,   2,   0,
 900      31,  20,   5,  15,   3,   0,   0,   0,   7,   3,   0,
 910      31,  26,   6,  15,   9,  14,   2,   8,   7,   3,   0}
 920 m_vset(78,v)
 930 /*
 940 v={
 950 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  3
 960      57,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
 970 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 980      31,   0,   0,   0,   0,  25,   0,   2,   0,   0,   0,
 990      31,   0,   0,   0,   0,  25,   0,   2,   0,   0,   0,
1000      31,   0,   0,   0,   0,  35,   0,   4,   0,   0,   0,
1010      31,  16,   0,  15,   3,   0,   0,   2,   0,   0,   0}
1020 m_vset(79,v)
1030 /*
1040 v={
1050 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
BassDrum
1060      44,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
1070 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
1080      31,  25,  25,  15,   5,   5,   5,   7,   5,   3,   5,
1090      31,  15,  15,  15,   5,   5,   5,   1,   5,   3,   5,
1100      31,  25,  25,  15,   5,   5,   5,   0,   5,   3,   5,
1110      31,  15,  15,  15,   5,   5,   5,   1,   5,   3,   5}
1120 m_vset(80,v)
1130 /*
1140 v={
1150 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  4
1160      57,  15,   0,   0,   0,   0,   0,   0,   0,   3,   0,
1170 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
```

▶メガドライブのソーサリアンはすごくよくできている。ノーマルX1しか持っていない僕にとってはすごく楽しめた。ちなみにX1ソーサリアンをX1Fで走らせてみると,「このプログラムはX1turboセンヨウデス」と出てくる。　松田　徹（14）奈良県

```
 1180      31,  0,  0,  0,  0, 24,  0,  2,  0,  0,  0,
 1190      31,  0,  0,  0,  0, 24,  0,  2,  0,  0,  0,
 1200      31,  0,  0,  0,  0, 36,  0,  4,  0,  0,  0,
 1210      18, 15,  0,  4,  3,  2,  0,  2,  0,  0,  0}
 1220 m_vset(81,v)
 1230 /*
 1240 v={
 1250 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  5
 1260      57, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
 1270 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 1280      31,  0,  0,  0,  0, 25,  0,  2,  0,  0,  0,
 1290      31,  0,  0,  0,  0, 25,  0,  2,  0,  0,  0,
 1300      31,  0,  0,  0,  0, 35,  0,  4,  0,  0,  0,
 1310      31, 16,  0,  3,  3,  0,  0,  2,  0,  0,  0}
 1320 m_vset(82,v)
 1330 /*
 1340 v={
 1350 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC6･･･
 1360      57, 15,  0,  0,  0,  0,  0,  0,  0,  3,  0,
 1370 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 1380      31,  0,  0,  0,  0, 25,  0,  4,  0,  0,  0,
 1390      31,  0,  0,  0,  0, 25,  0,  2,  0,  0,  0,
 1400      31,  0,  0,  0,  0, 35,  0,  4,  0,  0,  0,
 1410      27, 15,  9, 15,  3,  0,  0,  2,  0,  0,  0}
 1420 m_vset(83,v)
 1430 v={
 1440 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  7
 1450      57, 15,  3,  1,237,  0, 88,  0,  1,  3,  0,
 1460 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 1470      31,  0,  0, 15,  0, 24,  0,  2,  0,  0,  0,
 1480      31,  0,  0, 15,  0, 26,  0,  2,  0,  0,  0,
 1490      31,  0,  0, 15,  0, 34,  0,  4,  0,  0,  0,
 1500      31, 15,  2, 15,  3,  0,  0,  2,  0,  0,  0}
 1510 m_vset(84,v)
 1520 v={
 1530 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  8
 1540      57, 15,  3,  1,237,  0, 88,  0,  1,  3,  0,
 1550 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 1560      31,  0,  0,  0,  0, 25,  0,  1,  0,  0,  0,
 1570      31,  0,  0,  0,  0, 25,  0,  1,  0,  0,  0,
 1580      31,  0,  0,  0,  0, 35,  0,  2,  0,  0,  0,
 1590      31, 15,  4,  5,  3,  5,  0,  1,  0,  0,  0}
 1600 m_vset(85,v)
 1610 v={
 1620 /*  AF   OM   WF   SY   SP  PMD  AMD  PMS  AMS  PAN
VRC  9
 1630      57, 15,  3,  1,237,  0, 88,  0,  1,  3,  0,
 1640 /*  AR   DR   SR   RR   SL   OL   KS   ML  DT1  DT2  AME
 1650      31,  0,  0,  0,  0, 25,  0,  2,  0,  0,  0,
 1660      31,  0,  0,  0,  0, 25,  0,  2,  0,  0,  0,
 1670      31,  0,  0,  0,  0, 35,  0,  4,  0,  0,  0,
 1680      31, 15,  9,  8,  3,  3,  0,  2,  0,  0,  0}
 1690 m_vset(86,v)
 1700 /*
 1710 m_init():for i=1 to 8:m_alloc(i,8000):next
 1720 for i=1 to 8:m_assign(i,i):next
 1730 str a[256],b[256],c[256],d[256],e[256],f[256]
 1740 str g[256],h[256],j[256],k[256],l[256],m[256]
 1750 str n[256],o[256],p[256],q[256],r[256],s[256]
 1760 str t[256],u[256],w[256],x[256],y[256],z[256]
 1770 str aa[256],bb[256],cc[256],dd[256],ee[256],ff[256]
 1780 str gg[256],hh[256],jj[256],kk[256],ll[256],mm[256]
 1790 str nn[256],oo[256],pp[256],qq[256],rr[256],ss[256]
 1800 str tt[256],uu[256],ww[256],xx[256],yy[256],zz[256]
 1810 str sd[256],bd[256],t1[256],t2[256],t3[256],t4[256]
 1820 key 11,"y2,1":key 12,"y2,2"
 1830 key 19,"m_tempo(200)":key 20,"m_tempo(135)"+chr$(13)
 1840 /*
 1850 /*
 1860 a="t135  [d.c.]        r2 [coda] o2 116 @71 p3 q8  v14  y4
8,00 y15,0
 1870 b="dddd<dd>dd8<d>dd<d>ddd8<d>dd<d>ddd8<d>ddea<ea>
 1880 c="dddd<dd>dd8<d>dd<d>ddd8<d>dd<d>ddd8<d>dd@72ev10p1r@71p3
v15a2..
 1890 d="@72dv10p1r@71p3q7v14dd<q8@72dv10p1r@71p3q7v14d>q8@72dv1
0p1r@71p3q7v14d<d8>dd<d>de8<q8@72e8.v10p1r@71p3q7v14e>q8@72e-v10
p1r@71p3q7v14e-<e-8>e-e-<e->e-
 1900 e="@72dv10p1r@71p3q7v14dd<q8@72dv10p1r@71p3q7v14d>q8@72dv1
0p1r@71p3q7v14d<q8@72dv10p1r@71p3q7v14>dd<d>dq8@72ev10p1r@71p3q7
v14<q8@72e8.v10p1r@71p3q7v14e>q8@72ev10p1r@71p3q7v14e<q8@72ev10p
1r@71p3q7v14>aa<a>a
 1910 f="@72dv10p1r@71p3q7v14dd<q8@72dv10p1r@71p3q7v14d>q8@72dv1
0p1r@71p3q7v14d<d8>dd<d>de8<q8@72e8.v10p1r@71p3q7v14e>q8@72av10p
1r@71p3q7v14a<a>aaa<a>a
 1920 g="b-b-b-<q8@72b-v10p1r@71p3q7v14>b-<q8@72b-8v10p1r@71p3q7
v14q8@72b-v10p1r@71p3q7v14q8@72fv10p1r@71p3q7v14q8@72dv10p1r@71p
3q7v14>b-aaa<q8@72av10p1r@71p3q7v14>a<q8@72a8v10p1r@71p3q7v14>a<
d>a<fd>a<f>
 1930 h="a-a-a-<q8@72a-v10p1r@71p3q7v14>a-<q8@72a-8v10p1r@71p3q7
v14dfa-ba-fa-
 1940 j="q8@74@l1a32&aa-gg-fe>l16q7aa<
 1950 k="q8@74@l1a32&aa-gg-fe>l16q7aq8<
 1960 l="@75>>d4...p1v12r32p3v14e4...p1v12r32p3v14a4...p1v12r32p
3v14a-4...p1v12r32p3v14g4...p1v12r32p3v14@71a-4.<ea->@72a2v10p1r
p3v14@71<aa<a>a<a>a<a>
 1970 m="@72dp1v10rp3v14dp1v10rp3v14<dp1v10rp3v14>q7@71cq8@72cv1
0p1rp3v14q7@71ccc<q8@72cv10p1rp3v14q7@71>c<c>>q8@72bv10p1rp3v14b
v10p1rp3v14<bv10p1rp3v14q7@71>b-q8@72b-v10p1rp3v14q7@71b-<b->b-<
cc<c>c
 1980 n="q7@71d<d>@72dp1v10rp3v14<dp1v10rp3v14>q7@71cq8@72cv10p1
rp3v14q7@71ccc<q8@72cv10p1rp3v14q7@71>c<c>>b<b>q8@72bv10p1rp3v14
<bv10p1rp3v14q7@71>b-q8@72b-v10p1rp3v14q7@71b-<q8@72b-v10p1rp3v1
4q7@71>b-<b->q8@72b-v10p1rp3v14q7@71
 1990 o="q8@72av10p1rp3v14av10p1rp3v14q7@71<aq8@72av10p1rp3v14>a
v10p1rp3v14av10p1rp3v14q7@71aq8@72<av10p1rp3v14av10p1rp3>v14av10
p1rp3v14av10p1rp3v14q7@71<aq8@72av10p1rp3>v14av10p1rp3v14@71<<ag
fedd->a
 2000 z="[*]
 2010 /*
 2020 /*
 2030 m_trk(1,a)/*久々の登場でいささか緊張している、立川でござ
 2040 m_trk(1,b)/*います。今回はファミコンで最高の曲を揃えてい
 2050 m_trk(1,c)/*る「悪魔城伝説」より一面のテーマを作ったので
 2060 m_trk(1,d)/*とりあえずは聞いてみてください。こいつはコナ
 2070 m_trk(1,e)/*ミの技術を結集した（と思われる）ＶＲＣ６とい
 2080 m_trk(1,d)/*うＳＣＣもどきのスバラシイ音源を積んでいて、
 2090 m_trk(1,f)/*とてもファミコンとは信じられないような音を出
 2100 m_trk(1,g)/*しています。軽い気持ちで買ってきたＲＯＭから
 2110 m_trk(1,h)/*この曲が流れてきたときは自分の耳を疑ったくら
 2120 m_trk(1,j)/*いです。
 2130 m_trk(1,j)/*　どしぇすぽーん。
 2140 m_trk(1,j)/*　そんでもって、感動のあまり勢いだけで作って
 2150 m_trk(1,j)/*しまったのがこのプログラムなのであります。
 2160 m_trk(1,k)/*　だからして、
 2170 m_trk(1,k)/*１．もしかしたらあちこち音が違っている
 2180 m_trk(1,l)/*２．ハイハットのパートを手抜きしている
 2190 m_trk(1,m)/*３．音がうすっぺらい
 2200 m_trk(1,n)/*４．リストがぐちょんぐちょんである
 2210 m_trk(1,o)/*５．そのせいで一部テンポが遅れるところがある
 2220 m_trk(1,z)/*６．実際のゲームとは曲のバージョンが違う
 2230 /*
 2240 /*
 2250 a="      [d.c.]        r2 [coda] o2 116 @71 p3 q8  v14  y4
9,28
 2260 b="dddd<dd>dd8<d>dd<d>ddd8<d>dd<d>ddd8<d>ddea<ea>
 2270 c="dddd<dd>dd8<d>dd<d>ddd8<d>dd<d>ddd8<d>dd@72ev10p2r@71p3
v15a2..
 2280 d="@72dv10p2r@71p3q7v14dd<q8@72dv10p2r@71p3q7v14d>q8@72dv1
0p2r@71p3q7v14d<d8>dd<d>de8<q8@72e8.v10p2r@71p3q7v14e>q8@72e-v10
p2r@71p3q7v14e-<e-8>e-e-<e->e-
 2290 e="@72dv10p2p2r@71p3q7v14@71p3q7v14dd<q8@72dv10p2r@71p3q7v
14d>q8@72dv10p2r@71p3q7v14d<q8@72dv10p2r@71p3q7v14>dd<d>dq8@72ev
10r<q8@72e8.v10p2r@71p3q7v14e>q8@72ev10p2r@71p3q7v14e<q8@72ev10p
2r@71p3q7v14>aa<a>a
 2300 f="@72dv10p2r@71p3q7v14dd<q8@72dv10p2r@71p3q7v14d>q8@72dv1
0p2r@71p3q7v14d<d8>dd<d>de8<q8@72e8.v10p2r@71p3q7v14e>q8@72av10p
2r@71p3q7v14a<a>aaa<a>a
 2310 g="b-b-b-<q8@72b-v10p2r@71p3q7v14>b-<q8@72b-8v10p2r@71p3q7
v14q8@72b-v10p2r@71p3q7v14q8@72fv10p2r@71p3q7v14q8@72dv10p2r@71p
3q7v14>b-aaa<q8@72av10p2r@71p3q7v14>a<q8@72a8v10p2r@71p3q7v14>a<
d>a<fd>a<f>
 2320 h="a-a-a-<q8@72a-v10p2r@71p3q7v14>a-<q8@72a-8v10p2r@71p3q7
v14dfa-ba-fa-
 2330 j="q8@74@l1a32&aa-gg-fe>l16q7aa<
 2340 k="q8@74@l1a32&aa-gg-fe>l16q7aq8<
 2350 l="@75>>d4...p2v12r32p3v14e4...p2v12r32p3v14a4...p2v12r32p
3v14a-4...p2v12r32p3v14g4...p2v12r32p3v14@71a-4.<ea->@72a2v10p2r
p3v14@71<aa<a>a<a>a<a>
 2360 m="@72dp2v10rp3v14dp2v10rp3v14<dp2v10rp3v14>q7@71cq8@72cv1
0p2rp3v14q7@71ccc<q8@72cv10p2rp3v14q7@71>c<c>>q8@72bv10p2rp3v14b
v10p2rp3v14<bv10p2rp3v14q7@71>b-q8@72b-v10p2rp3v14q7@71b-<b->b-<
cc<c>c
 2370 n="q7@71d<d>@72dp2v10rp3v14<dp2v10rp3v14>q7@71cq8@72cv10p2
rp3v14q7@71ccc<q8@72cv10p2rp3v14q7@71>c<c>>b<b>q8@72bv10p2rp3v14
<bv10p2rp3v14q7@71>b-q8@72b-v10p2rp3v14q7@71b-<q8@72b-v10p2rp3v1
4q7@71>b-<b->q8@72b-v10p2rp3v14q7@71
 2380 o="q8@72av10p2rp3v14av10p2rp3v14q7@71<aq8@72av10p2rp3v14>a
v10p2rp3v14av10p2rp3v14q7@71aq8@72<av10p2rp3v14av10p2rp3>v14av10
p2rp3v14av10p2rp3v14q7@71<aq8@72av10p2rp3>v14av10p2rp3v14@71<<ag
fedd->a
 2390 /*
 2400 /*
 2410 m_trk(2,a)/*といった点は見て見ぬふりもとい、聞いて聞かぬ
 2420 m_trk(2,b)/*ふりで軽く流して下さい。
 2430 m_trk(2,c)/*　とはいってもＶＲＣ６の音色をＦＭ音源で再現
 2440 m_trk(2,d)/*するのにかなり苦労したので、それなりの聞きご
 2450 m_trk(2,e)/*たえはあると思います。
 2460 m_trk(2,d)/*　ところで今回のリストはご覧のとおり、今まで
 2470 m_trk(2,f)/*にもましてぬちょぬちょしています。Ｙコマンド
 2480 m_trk(2,g)/*によるソフトＬＦＯ、リリースレートを操作して
 2490 m_trk(2,h)/*の疑似リバーブ効果、パンポットを振っての音場
 2500 m_trk(2,j)/*効果など、ＭＭＬが汚くなる要因が重なっている
 2510 m_trk(2,j)/*からです。仰々しいことをやっているわりにはそ
 2520 m_trk(2,j)/*んなに効果は上がっていなかったりするので、こ
 2530 m_trk(2,j)/*こはひとつ、「はなうたふふふん」で入力してく
 2540 m_trk(2,k)/*ださいませ。
 2550 m_trk(2,k)/*　さて、なんだか催促されているようなので次回
 2560 m_trk(2,l)/*より、「立川君のよいこのためのＦＭ音源講座」
 2570 m_trk(2,m)/*を再開しよーかなと考えていたりします。あまり
 2580 m_trk(2,n)/*期待しないで待っていて下さいね。
 2590 m_trk(2,o)/*　ではこのプログラムに関する文句、禁句、俳句
 2600 m_trk(2,z)/*をお待ち申し上げております。では。　立川正之
 2610 /*
 2620 /*
 2630 a="      [d.c.]        r2 [coda] o3 116 @73 p3 q8 @v127 y5
0,20
 2640 b="r4d8.c4&@l1>bb-aa-gg-fee-dd-cy8,2<l16c8.>b4&@l1b-aa-gg-
fee-dd-c>b<y8,2l16b8.b-8.agb-a<d-e8.&@l1ee-dd-c>bb-aa-gg-fl16e<<
 2650 c="d8.c4&@l1>bb-aa-gg-fee-dd-cy8,2<l16c8.>b4&@l1b-aa-gg-fe
e-dd-c>b<y8,2l16b8.b-8a<d-e<v14ged->aged->aged-v15b-agfev15
 2660 d="@76<fed>v14a4.v15a<defedaa-8.e8.a-8g8.e-8.g8
 2670 e="fed>v14a4.v15a<defedaa-8.e8.<d8d-de>a6...>
 2680 f="fed>v14a4.v15a<defedaa-8.<d8.f8c+8.e8.a8>
```

▶大阪が毎月20日がノーマイカーデーになったということを知っていますか。しかし4/20に車の交通量を調べたらしいんですが，なんと4/19と変わりはなかったそうです。それどころか逆に多かったところもあるそうです。　乾　義和 (19) 大阪府

```
 2690 g="@83fffd8d>b-8.<d8.f8g8f8.d8.>a8.a<d>a<fd>a8<fffd8d>a-8.
b<d>b<fd>b<dd-8.d8.e-8.e8.g8a8
 2700 h="@v127t132@84f8.d8.>a8a-8.<e8.d8d-4&@l1|:3y50,48d-y50,76
d-y50,104d-y50,132d-y50,104d-y50,76d-y50,48d-y50,20d-y50,248cy50
,220cy50,192cy50,164cy50,192cy50,220cy50,248cy50,20d-:|y50,20y8,
2
 2710 j="d4&|:3y50,48dy50,76dy50,104dy50,132dy50,104dy50,76dy50,
48dy50,20dy50,248d-y50,220d-y50,192d-y50,164d-y50,192d-y50,220d-
y50,248d-y50,20d:|y50,20y8,2>g8.b-8.<d8f8.e8.d8t133@76v14a16b-16
a16b-16a16b-16a16b-16v14t135a4&
 2720 k="|:3y50,48ay50,76ay50,104ay50,132ay50,104ay50,76ay50,48a
y50,20ay50,248a-y50,220a-y50,192a-y50,164a-y50,192a-y50,220a-y50
,248a-y50,20a:|y50,20y8,2@v126
 2730 l="@l1d16f16a16<d16&|:3y50,48dy50,76dy50,104dy50,132dy50,1
04dy50,76dy50,48dy50,20dy50,248d-y50,220d-y50,192d-y50,164d-y50,
192d-y50,220d-y50,248d-y50,20d:|y50,20y8,2>
 2740 m="l16dfa<d8.>dfa<d8.>eg<ce>
 2750 n="@l1@v127f16a16<d16f16&|:3y50,48fy50,76fy50,104fy50,132f
y50,104fy50,76fy50,48fy50,20fy50,248ey50,220ey50,192ey50,164ey50
,192ey50,220ey50,248ey50,20f:|y50,20y8,2>
 2760 o="l16fa<df8.>fa<df8.efga->v15|:8ab-:||:8a32b-32:|l32a<d-e
a&v15a&v14a&v13a&v12a>a<d-ea&v11a&v10a&v9a&v8a
 2770 /*
 2780 /*
 2790 m_trk(3,a)
 2800 m_trk(3,b)
 2810 m_trk(3,c)
 2820 m_trk(3,d)
 2830 m_trk(3,e)
 2840 m_trk(3,d)
 2850 m_trk(3,f)
 2860 m_trk(3,g)
 2870 m_trk(3,h)
 2880 m_trk(3,j)
 2890 m_trk(3,k)
 2900 m_trk(3,l)
 2910 m_trk(3,l)
 2920 m_trk(3,m)
 2930 m_trk(3,n)
 2940 m_trk(3,n)
 2950 m_trk(3,o)
 2960 m_trk(3,z)
 2970 /*
 2980 /*
 2990 a="       [d.c.]  @l1rr8r2 [coda] o3 l16 @73 p3 q8  v14  y5
1,40
 3000 b="r4d8.c4&@l1>bb-aa-gg-fee-dd-cy8,3<l16c8.>b4&@l1b-aa-gg-
fee-dd-c>b<y8,3l16b8.b-8.agb-a<d-e8.&@l1ee-dd-c>bb-aa-gg-fl16e<<
 3010 c="d8.c4&@l1>bb-aa-gg-fee-dd-cy8,3<l16c8.>b4&@l1b-aa-gg-fe
e-dd-c>b<y8,3l16b8.b-8a<d-e<v13ged->aged->aged-v14b-agfev13
 3020 d="@76<fed>v12a4.v13a<defedaa-8.e8.a-8g8.e-8.g8
 3030 e="fed>v12a4.v13a<defedaa-8.e8.<d8d-de>a6...>
 3040 f="fed>v12a4.v13a<defe@82>v15l32
 3050 g="e.v12p1r64p3v15g-.v12p2r64p3v15a-.v12p1r64p3v15a.v12p2r
64p3v15b.v12p1r64p3<v15d-.v12p2r64p3v15d.v12p1r64p3>v14b.v11p2r6
4p3v15<
 3060 h="a.v12p1r64p3v15g.v12p2r64p3v15f.v12p1r64p3v15e.v12p2r64
p3v15d.v12p1r64p3v15d-.v12p2r64p3>v15b-.v12p1r64p3v15a.v12p2r8r6
4p3@83q8v12<
 3070 j="l16fffd8d>b-8.<d8.f8g8f8.d8.>a8.a<d>a<fd>a8<fffd8d>a-8.
b<d>b<fd>b<dd-8.d8.e-8.e8.g8a8.
 3080 k="@84f8.d8.>a8a-8.<e8.d8d-4&@l1|:3y51,68d-y51,96d-y51,124
d-y51,152d-y51,124d-y51,96d-y51,68d-y51,40d-y51,12d-y51,240cy51,
212cy51,184cy51,212cy51,240cy51,12d-y51,40d-:|y51,40y8,3
 3090 l="d4&|:3y51,68dy51,96dy51,124dy51,152dy51,124dy51,96dy51,
68dy51,40dy51,12dy51,240d-y51,212d-y51,184d-y51,212d-y51,240d-y5
1,12dy51,40d:|y51,40y8,3>g8.b-8.<d8f8.e8.d8@76v11a16b-16a16b-16a
16b-16a16b-16v12a4&
 3100 m="|:2y51,68ay51,96ay51,124ay51,152ay51,124ay51,96ay51,68a
y51,40ay51,12ay51,240a-y51,212a-y51,184a-y51,212a-y51,240a-y51,1
2ay51,40a:|a48y51,40v13
 3110 n="@l1d16f16a16<d16&|:3y51,68dy51,96dy51,124dy51,152dy51,1
24dy51,96dy51,68dy51,40dy51,12dy51,240d-y51,212d-y51,184d-y51,21
2d-y51,240d-y51,12dy51,40d:|y51,40y8,3>
 3120 o="l16dfa<d8.>dfa<d8.>eg<ce>
 3130 p="@l1v14f16a16<d16f16&|:3y51,68fy51,96fy51,124fy51,152fy5
1,124fy51,96fy51,68fy51,40fy51,12fy51,240ey51,212ey51,184ey51,21
2ey51,240ey51,12ey51,40f:|y51,40y8,3>
 3140 q="l16fa<df8.>fa<df8.efga->v13|:8ab-:||:8a32b-32:|l32a<d-e
a&v12a&v11a&v10a&v9a>a<d-ea&v8a&v7a&v6a&v5a
 3150 /*
 3160 /*
 3170 m_trk(4,a)
 3180 m_trk(4,b)
 3190 m_trk(4,c)
 3200 m_trk(4,d)
 3210 m_trk(4,e)
 3220 m_trk(4,d)
 3230 m_trk(4,f)
 3240 m_trk(4,g)
 3250 m_trk(4,h)
 3260 m_trk(4,j)
 3270 m_trk(4,k)
 3280 m_trk(4,l)
 3290 m_trk(4,m)
 3300 m_trk(4,n)
 3310 m_trk(4,n)
 3320 m_trk(4,o)
 3330 m_trk(4,p)
 3340 m_trk(4,p)
 3350 m_trk(4,q)
 3360 m_trk(4,z)
 3370 /*
 3380 /*
 3390 a="       [d.c.]          r2 [coda] o3 l16 @79 p3 q8  v15  y5
2,08
 3400 b="r4a8.g4&@l1g-fee-dd-c>bb-aa-gy8,4<l16g8.g4&@l1g-fee-dd-
c>bb-aa-g<y8,4l16g8.g8.fegfa<d-8.&@l1c>bb-aa-gg-fee-dd-c>bb-aa-g
g-fee-dd-l16y8,4<
 3410 c="a8.g4&@l1g-fee-dd-c>bb-aa-gy8,4<l16g8.g4&@l1g-fee-dd-c>
bb-aa-g<y8,4l16g8.>v11g8a<d-e<ged->aged->aged-b-agfe<
 3420 d="@81v12a2&a8.v9p1rp3v12<d8.v9p2rp3v12e8 v9p1rp3v12>b4 v9
p2rp3v12<e-8v9p1rp3v12>b-4v9p2rp3v12a2&a8.v9p1rp3v12<d8.v9p2rp3v
12e8. v9p1rp3v12>b4.. v9p2rp3v12<d-8.v9p1rp3v12>
 3430 e="@81v12a2&a8.v9p2rp3v12<d8.v9p1rp3v12e8 v9p2rp3v12>b4 v9
p1rp3v12<e-8v9p2rp3v12>b-4v9p1rp3v12a2&a8.v9p2rp3v12<d8.v9p1rp3v
12 a-4..v9p2rp3v12a4..v9p1rp3v14
 3440 f="d8v11p1rp3v14>b-8v11p1rp3v14f8.v11p1rp3v14fv11p1rp3v14b
-v11p1rp3v14<dv11p1rp3v14d8v11p1rp3v14>a8v11p1rp3v14f6...
 3450 g="v11p1rp3v14a8.v11p1rp3v14<d8v11p1rp3v14>b8v11p1rp3v14a-
2&a-v11p1rp3v14<d-8v11p1rp3v14d8v11p1rp3v14e-8v11p1rp3v14e8v11p1
rp3v14gv11p1rp3v14av11p1rp3
 3460 h="@76v13d8.>a8.f8d2g8.a8.g8f8.e8.d8g4.b-8b2<v14d4.edd-2>
 3470 j="@79v15dafadafacafacafa>b<afa>b<afa>b-<fdfcgegdafa<d>afa
cafa<d>afa>b<afa<d>afa>b-<a<afdfa<d>
 3480 k="v15q6@86f<f>f<f>ffff<f>f<f>fffffq8@79e4f4l32agfedc>bagf
edc>bag
 3490 /*
 3500 /*
 3510 m_trk(5,a)
 3520 m_trk(5,b)
 3530 m_trk(5,c)
 3540 m_trk(5,d)
 3550 m_trk(5,e)
 3560 m_trk(5,f)
 3570 m_trk(5,g)
 3580 m_trk(5,h)
 3590 m_trk(5,j)
 3600 m_trk(5,k)
 3610 m_trk(5,z)
 3620 /*
 3630 /*
 3640 a="       [d.c.]          r2 [coda] o3 l16 @79 p3 q8  v15  y5
3,32
 3650 b="r4f8.e4&@l1e-dd-c>bb-aa-gg-fey8,5<l16e8.d4&@l1d-c>bb-aa
-gg-fee-d<y8,5l16d8.d8.dddd-eg8.&@l1g-fee-dd-c>bb-aa-gg-fee-dd-c
>bb-aa-gl16y8,5<<
 3660 c="f8.e4&@l1e-dd-c>bb-aa-gg-fey8,5<l16e8.d4&@l1d-c>bb-aa-g
g-fee-d<y8,5l16d8.>v9g8a<d-e<ged->aged->aged-b-agfe<
 3670 d="@81v12f2&f8.v9p1rp3v12 a8.v9p1rp3v12b8 v9p1rp3v12 a-4v9
p1rp3v12 b-8v9p1rp3v12 g4 v9p1rp3v12f2&f8.v9p1rp3v12 a8.v9p1rp3v
12b8. v9p1rp3v12 a-4..v9p1rp3v12 a8. v9p1rp3v12
 3680 e="@81v12f2&f8.v9p1rp3v12 a8.v9p1rp3v12b8 v9p1rp3v12 a-4v9
p1rp3v12 b-8v9p1rp3v12 g4 v9p1rp3v12f2&f8.v9p1rp3v12 a8.v9p1rp3v
12 e4.. v9p1rp3v12e4..v9p2rp3v13
 3690 f="b-8v9p2rp3v13f8v9p2rp3v13d8.v9p2rp3v13dv9p2rp3v13fv9p2r
p3v13b-v9p2rp3v13a8v9p2rp3v13f8v9p2rp3v13d6...v9p2rp3v13f8.v9p2r
p3v13b8v9p2rp3v13f8v9p2rp3v13f2&fv9p2rp3v13a8v9p2rp3v13a8v9p2rp3
v13a8v9p2rp3v13a8v9p2rp3v13av9p2rp3v13av9p2rp3v14
 3700 f="b-8v11p2rp3v14f8v11p2rp3v14d8.v11p2rp3v14dv11p2rp3v14fv
11p2rp3v14b-v11p2rp3v14a8v11p2rp3v14f8v11p2rp3v14d6...
 3710 g="v11p2rp3v14f8.v11p2rp3v14b8v11p2rp3v14f8v11p2rp3v14f2&f
v11p2rp3v14a8v11p2rp3v14a8v11p2rp3v14a8v11p2rp3v14a8v11p2rp3v14a
v11p2rp3v14av11p2rr8p3v14
 3720 k="@76v10<d8.>a8.f8d2g8.a8.g8f8.e8.d8g4.b-8b2<v11d4.edd-4.
.>
 3730 l="@79dafadafacafacafa>b<afa>b<afa>b-<fdfcgegdafa<d>afacaf
a<d>afa>b<afa<d>afa>b-<a<afdfa
 3740 m="v15q6@86d-<d->d-<d->d-d-d-d-<d->d-<d->d-d-d-d-d-q8@79d-
4d4r2
 3750 /*
 3760 /*
 3770 m_trk(6,a)
 3780 m_trk(6,b)
 3790 m_trk(6,c)
 3800 m_trk(6,d)
 3810 m_trk(6,e)
 3820 m_trk(6,f)
 3830 m_trk(6,g)
 3840 m_trk(6,k)
 3850 m_trk(6,l)
 3860 m_trk(6,m)
 3870 m_trk(6,z)
 3880 /*
 3890 /*
 3900 a="       [d.c.]  @l2rr8r2 [coda] o3 l16 @79 p3 q8  v13  y5
4,24
 3910 b="r4a8.g4&@l1g-fee-dd-c>bb-aa-gy8,6<l16g8.g4&@l1g-fee-dd-
c>bb-aa-g<y8,6l16g8.g8.fegfa<d-8.&@l1c>bb-aa-gg-fee-dd-c>bb-aa-g
g-fee-dd-l16y8,6<
 3920 c="a8.g4&@l1g-fee-dd-c>bb-aa-gy8,6<l16g8.g4&@l1g-fee-dd-c>
bb-aa-g<y8,6l16g8.>v5g8a<d-e<ged->aged->aged-b-ag<
 3930 d="@81v12d2&d8.v9p2rp3v12 f8.v9p2rp3v12a-8v9p2rp3v12 e4 v9
p2rp3v12 g8 v9p2rp3v12 e-4v9p2rp3v12d2&d8.v9p2rp3v12 f8.v9p2rp3v
12a-8.v9p2rp3v12 e4 @82v14av12p1rp3v14a-gv12p2rp3v14g-fe
 3940 e="@81v12d2&d8.v9p2rp3v12 f8.v9p2rp3v12a-8v9p2rp3v12 e4 v9
p2rp3v12 g8 v9p2rp3v12 e-4v9p2rp3v12d2&d8.v9p2rp3v12 f8.v9p2rp3@
82v12
 3950 f="l32e16..v9p2r64p3v12g-.v9p1r64p3v12a-.v9p2r64p3v12a.v9p
1r64p3v12b.v9p2r64p3<v12d-.v9p1r64p3v12d.v9p2r64p3>v11b.v8p1r64p
3v12<a.v9p2r64p3v12g.v9p1r64p3v12f.v9p2r64p3v12e.v9p1r64p3v12d.v
9p2r64p3v12c.v9p1r64p3>v12b.v9p2r64p3v14@82l32f.v12p1r64p3
 3960 g="b-.v12p2r64p3v14<d.v12p1r64p3v14>b-.v12p2r64p3v14f.v12p
1r64p3v14b-.v12p2r64p3v14<f.v12p1r64p3v14>b-.v12p2r64p3v14f.v12p
1r64p3v14b-.v12p2r64p3v14<d.v12p1r64p3v14f.v12p2r64p3v14b-.v12p1
r64p3v14f.v12p2r64p3v14d.v12p1r64p3v14>b-.v12p2r64p3v14
 3970 h="d.v12p1r64p3v14f.v12p2r64p3v14d.v12p1r64p3v14a.v12p2r64
p3v14<d.v12p1r64p3v14>a.v12p2r64p3v14f.v12p1r64p3v14a.v12p2r64p3
```

▶Z80's Barのイッパツマンの替え歌ですが,「4メガのはやいスピード」の前に1小節入るんじゃないでしょうか。　安川　実 (16) 愛知県

```
v14d.v12p1r64p3v14f.v12p2r64p3v14a.v12p1r64p3v14f.v12p2r64p3v14<
d.v12p1r64p3v14>a.v12p2r64p3v14f.v12p1r64p3v14d.v12p2r64p3
 3980 j="v14f.v12p1r64p3v14d.v12p2r64p3v14f.v12p1r64p3v14f.v12p2
r64p3v14b.v12p1r64p3v14<d.v12p2r64p3v14>f.v12p1r64p3v14b.v12p2r6
4p3v14<d.v12p1r64p3v14>b.v12p2r64p3v14b.v12p1r64p3v14f.v12p2r64p
3v14<d.v12p1r64p3v14>b.v12p2r64p3v14f.v12p1r64p3v14b.v12p2
 3990 k="r64l16p3v15@79a8.b8.<c8.d-8.e8e8@l1@85<<@v127
 4000 l="y54,20by54,200b-y54,100b-y54,20b-y54,200ay54,100ay54,20
ay54,200a-y54,100a-y54,20a-y54,200gy54,100gy54,20gy54,200g-y54,1
00g-y54,20g-y54,200fy54,100fy54,20fy54,200ey54,100ey54,20ey54,20
0e-y54,100e-y54,20e-y54,200dy54,100d
 4010 m="y54,0dy54,100d-y54,0d-y54,100cy54,0c>
 4020 n="y8,6l16y54,24<<<@79v14av13av12av14av13a<v14egafv13fv12f
v14dv13dv12d>v14av13av14gv13gv12gv14gv13gv14b-<df>a-v13a-v12a-v1
4a-v13a-<v14ea-v13a-v12a-v11a-v10a-v9a-v8a-v7a-v6a-v5a-v4a-v3a-v
2a-v1a-v0a-r8.
 4030 o="v14>dfa<d6...>dfa<d6...>dfa<d8.>dfa<d8.>eg<cev15dfa<d6.
..>dfa<d6...>dfa<d8.>dfa<d8.d-ded
 4040 p="v15q6@86>>a<a>a<a>aaaa<a>a<a>aaaaaq8@79a4a4r2r8
 4050 /*
 4060 /*
 4070 m_trk(7,a)
 4080 m_trk(7,b)
 4090 m_trk(7,c)
 4100 m_trk(7,d)
 4110 m_trk(7,e)
 4120 m_trk(7,f)
 4130 m_trk(7,g)
 4140 m_trk(7,h)
 4150 m_trk(7,j)
 4160 m_trk(7,k)
 4170 m_trk(7,l)
 4180 m_trk(7,m)
 4190 m_trk(7,l)
 4200 m_trk(7,m)
 4210 m_trk(7,l)
 4220 m_trk(7,m)
 4230 m_trk(7,l)
 4240 m_trk(7,m)
 4250 m_trk(7,l)
 4260 m_trk(7,m)
 4270 m_trk(7,l)
 4280 m_trk(7,m)
 4290 m_trk(7,n)
 4300 m_trk(7,o)
 4310 m_trk(7,p)
 4320 m_trk(7,z)
 4330 /*
 4340 /*
 4350 a="      [d.c.]                   o3 l16 @77 p3 q8 @v127 y5
5,16
 4360 b="y3,3y2,1@80>>c<<@77y2,2a32&y2,2a32y2,2ay2,2ay2,2ay2,2ay
2,2ay2,2a[coda]
 4370 c="y2,1@80>>cy2,1cy2,1cy2,1c<<@77
 4380 d="y2,2@77a@80>>cy2,1cy2,2<<@78a@80>>cy2,1cy2,1cy2,1c<<
 4390 e="y2,2@77a32&y2,2a32y2,2ay2,2@78a&y2,2a@77
 4400 f="y2,2@77a8y2,2a&y2,2aa8a8y2,2r32y2,2r32y2,2ry2,2ry2,2ry2
,2ry2,2ry2,2ry2,2r
 4410 g="@80>>y2,1cry2,1cy2,1c<<@77y2,2araa@80>>y2,1cry2,1cy2,1c
<<y2,2@77a>>@80cy2,1cy2,1cy2,1cry2,1cy2,1cy2,2cr<<@77aa>>@80y2,1
cry2,2cc<<@77y2,2ay2,2ry2,2aa
 4420 h="@80>>y2,1cy2,1cy2,1cy2,2<<@77a8@80>>y2,1cy2,2<<@77a8a@8
0>>y2,1cy2,2<<@77a8a32a32y2,2ay2,2ay2,2a
 4430 j="@77y2,2a@80>>y2,1cy2,1c<<":j=j+j+j+j+"@77y2,2a@80>>y2,1
c<<@77y2,2a@80>>y2,1c<<
 4440 k="@80>>y2,1crry2,1c<<@77ar>>@80y2,1cy2,1cy2,2@77<<a4a4
 4450 l="@80>>y2,1crry2,1c<<@77ay2,2r>>@80y2,1cy2,1c@77<<ay2,2a3
2&y2,2a32&y2,2a&y2,2a&y2,2a&y2,2a&y2,2a&y2,2a
 4460 m="@80>>y2,1cry2,1cry2,2rry2,1cy2,1c<<@77a@80>>y2,1cy2,1cy
2,1cy2,2rry2,1cy2,1c<<
 4470 n="@80>>y2,1cry2,1cry2,2rry2,1cy2,1c<<@77a@80>>y2,1cy2,2cy
2,1cy2,2ry2,2ry2,2cy2,1c<<
 4480 o="@80>>y2,1cry2,1cry2,2<<@77ay2,2ray2,1@80>>c<<@77ay2,1@8
0>>c@77<<ay2,1cy2,2ary2,2ary2,1@80>>cry2,1cry2,2@77<<ay2,2ra@80>
>y2,1cry2,2@77<<a32y2,2a32y2,2ry2,2ry2,2ry2,2ry2,2ry2,2r
 4490 /*
 4500 /*
 4510 m_trk(8,a)
 4520 m_trk(8,b)
 4530 m_trk(8,c)
 4540 m_trk(8,d)
 4550 m_trk(8,d)
 4560 m_trk(8,d)
 4570 m_trk(8,e)
 4580 m_trk(8,c)
 4590 m_trk(8,d)
 4600 m_trk(8,d)
 4610 m_trk(8,d)
 4620 m_trk(8,f)
 4630 m_trk(8,g)
 4640 m_trk(8,g)
 4650 m_trk(8,g)
 4660 m_trk(8,g)
 4670 m_trk(8,h)
 4680 m_trk(8,h)
 4690 m_trk(8,h)
 4700 m_trk(8,j)
 4710 m_trk(8,k)
 4720 m_trk(8,k)
 4730 m_trk(8,k)
 4740 m_trk(8,l)
 4750 m_trk(8,m)
 4760 m_trk(8,n)
 4770 m_trk(8,m)
 4780 m_trk(8,n)
 4790 m_trk(8,o)
 4800 m_trk(8,z)
 4810 /*
 4820 /*
 4830 m_play()
```

リスト4　コンフィギュレーションファイル

```
1=bd1v12.pcm
2=sd3v13.pcm
```

日本音楽著作権協会(出)許諾第9070259-001号

リスト5　この木なんの木

```
  10 m_init()
  20 dim str a(15)[256]
  30 int i,j
  40 for i=1 to 8
  50 m_alloc(i,1000)
  60 m_assign(i,i)
  70 next
  80 a(0)="o4 @34 v14 t130 q4 p3 l8"
  90 a(1)="r1r1|:g<c4de4dc>fffga2g4<deffgagfeeerrr>g4<cdee4ef2..>ag4.<feed16r16
>b&|1<c1>:|<c2..>ag4.<feed16r16>b&<c1>"
 100 a(2)="o5 @25 v12 t130 q8 p3 l8"
 110 a(3)="r1r1|:r1r2aab<c>b1<c2.r4c2>b-2a2&afgag2f2|1e1:|e2&efgag2f2e1"
 120 a(4)="o5 @25 v12 t130 q8 p3 l8"
 130 a(5)="r1r1|:r1r2ffgag1g2.r4g2g2f2&fr4ag4.ggggg|1g1:|g2..ag4.gggggg1"
 140 a(6)=a(0)+"v13"+"p2"
 150 a(7)="r1r1|:o4g<c4de4dc>fffga2g4b<cddefedcccrrr>g4<cdee4ef2..>ag4.<aggf16r
16f&|1e1:|e2..>ag4.<aggf16r16f&e1"
 160 a(8)="o3 @14 v12 t130 q8 p3 l16"
 170 a(9)="r2r8gaged8@10c4c8.crrc8>gag8<|:4c4c8.crrc8>gag8f4f8.frrf8cdc8g4g8.gr
rg8ded8<c4c8.crrc8>gag8<:|>g4g8.grrg8ded8<c4c8.crrc8>gag8<c4"
 180 a(10)="o2          t130 q8 p3 l4"
 190 a(11)="r1|:19v12@48cv14@45cv12@48c8c8v14@45c:|c"
 200 a(12)="o4 @7 v08 t130 q8 p3 l16"
 210 a(13)="r1r4e8.erre4e8|:4e4e8.erre4e8f4f8.frrf4f8d4d8.drrd4d8e4e8.erre4e8:|
d4d8.drrd4d8e4e8.erre4e8e4@40t145l4g&<g>t130g-&<g->t85f&<f>t60e1"
 220 a(14)="o4 @7 v08 t130 q8 p3 l16"
 230 a(15)="r1r4g8.grrg4g8|:18g4g8.grrg4g8:|g4@40t145l4e&<e>t130e-&<e->t85d&<d>
t60c1"
 500 for j=0 to 7
 510 for i=0 to 1
 520 m_trk(j+1,a(j*2+i))
 530 next
 540 next
 550 m_play()
```

▶MZ-700のQDを売ってくださった細谷晴夫さん。住所がわからなくなってしまいました。送ってください。すみません。　原　知久（20）大阪府

BACK ISSUES バックナンバー案内

ここには1989年6月号から1990年5月号までをご紹介しました。現在1989年6～12,1990年1～5月号までの在庫がございます。バックナンバーおよび定期講読のお申し込み方法については,178ページを参照してください。

1989

6月号
特集　これからのXfamily
X68000に光磁気ディスクを/学習リモコンの製作
THE SOFTOUCH　ライトニングバッカス/Might and MagicII他
● OPMA用外部関数による　KENBAN.BAS
● X1/X1turbo用ドライブゲーム　Spirit of Rally
● X1turboZ用　これ,パズルなんですか。
MZ-2500 MIDI入門(1)MIDIボードを作る
C調言語講座PRO-68K/X68000マシン語プログラミング
全機種共通システム　超小型コンパイラTTC

7月号
特集 3Dグラフィックへの飛翔
Zバッファアルゴリズム/スムースシェイディング 他
THE SOFTOUCH Terazzo PRO-68K/アドヴァンスト・ファンタジアン
新連載
DōGA・CGアニメーション講座
MZ-2500用グラフィックエディタ作成講座
マシン語カクテル in Z80's Bar
X-BASICプログラミング調理実習
全機種共通システム TTC用パズルゲームTIC BAN
X68000マシン語プログラミング/C調言語講座PRO-68K 他

8月号
特集1　X1プログラミングガイドブック
PCGの基礎から奥義まで/超高速ラインルーチン 他
特集2　3Dグラフィックの深淵へ
スキャンラインZバッファ/3Dモデリング 他
新連載 (で)のショートプロぱーてぃ
X68000マシン語プログラミング/C調言語講座 PRO-68K
X-BASICプログラミング調理実習/DōGA・CGA講座
MZ-2500用グラフィックエディタ/Z80's Bar 他
全機種共通システム CP/M用ファイルコンバータ

9月号
特集　活用ハードディスク&プリンタ
各社ハードディスク接続総チェック/ハードディスク雑学講座/COPYキーメニュー/ビデオプリンタ活用プログラム　他
THE SOFTOUCH　ジェノサイド/琉球/mFORTH Compiler
● サイバースティックで遊ぶ　不思議な環境ソフトの世界
● X1/X1turbo用シューティングゲーム　Defeat X
Z80's Bar/MZ-2500グラフィックエディタ　他
[X68000] X-BASIC/マシン語/C調言語講座/DōGA・CGA
全機種共通システム　生物進化シミュレーションBUGS

10月号
特集　ゲーム面白心理学
ソーサリアン・宇宙からの訪問者/ファンタジーゾーン
ねじ式/ガウディ・バルセロナの風/サバッシュ　他
● MZ-700用シューティングゲームSide Roll-F
● X1/X1turdo用カードゲームBonding
ショートプロ/Z80's Bar/MZ-2500グラフィックエディタ
X68000マシン語/X-BASIC/C調言語講座/DōGA・CGA
THE SOFTOUCH　Z'sTRIPHONY DIGITAL CRAFT/James68K
全機種共通システム　小型インタプリタ言語TTI

11月号
特集　microComputer入門
初歩からのCPU物語/RISCプロセッサの設計と製作
X68000&X1で周辺LSIを使いこなそう
連載
ショートプロ/Z80's Bar/MZ-2500グラフィックエディタ
X68000マシン語/X-BASIC/C調言語講座/DōGA・CGA
● X68000用カードゲームばばぬき
LIVE in '89　メタルホーク/オブ・ラ・ディ,オブ・ラ・ダ
THE SOFTOUCH　Stationery PRO-68K/リングマスター1
全機種共通システム　TTI用パズルゲームPUSH BON!

12月号
特集　Cプログラミングへの招待
付録　C言語簡易リファレンス
連載
ショートプロぱーてぃ/Z80's Bar
X68000マシン語/X-BASIC/DōGA・CGA
● Oh!X2周年特別企画「素粒子の声が聞こえる」
● X1/turbo用アクションゲームACTIVE UNIT
LIVE in '89　天空の城ラピュタ/ギャラクシーフォース
THE SOFTOUCH　38万キロの虚空/た～みのる2
全機種共通システム　SLANG用リダイレクションライブラリ

1990

1月号
特集1　オペレーティングスタイルの研究
特集2　Cプログラミング応用編
連載
ショートプロぱーてぃ/Z80's Bar
X68000マシン語/C調言語講座/DōGA・CGA
● X1/turbo用シミュレーションゲームSuper Battle
LIVE in '90　さよならを過ぎて/RYDEEN
THE SOFTOUCH　レナム/メタルサイト
全機種共通システム　WORM KUN/再掲載SLANG
特別付録　X68000 THE SOFTWARE CATALOGUE

2月号
特集　画像圧縮へのアプローチ
連載
ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X68000マシン語/C調言語講座/X-BASIC調理実習
● X68000用ゲームプログラムGon Gon
● MZ-700用紙芝居Eyelarth
LIVE in '90　オーダイン/魔女の宅急便
THE SOFTOUCH　A-JAX/フラッピー2/夢幻戦士ヴァリスII
マジックパレット/Mu-1/CYBERNOTE PRO-68K
全機種共通システム　超小型コンパイラTTC++

3月号
特集　MUSICアドベンチャー
X68000用MIDIドライバ&音源エディタ
なんでも鳴らせるOPMD.X/MMLを楽譜データに
連載
ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
C調言語講座/X-BASIC調理実習
● X1/turboシミュレーションCRISIS in Tokyo
LIVE in '90　パワードリフト/スキーム/となりのトトロ
THE SOFTOUCH　ナイトアームズ/斬/ダンジョンマスター
全機種共通システム　超多機能アセンブラOHM-Z80

4月号
特集　ゲームシステム文学誌
1989年度GAME OF THE YEAR発表
連載
ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X-BASIC調理実習/C調言語講座/X68000マシン語
● X1・MZ-2000/2500用RPG The Cave of Dalk
● うわさの68040,ついに登場
LIVE in '90　バーニングフォース(OPMD対応)
THE SOFTOUCH　The Fille Professor/HOST PRO-68K
全機種共通システム　ファジィコンピュータシミュレータI-MY

5月号
特集　BASICプログラミング
第5回　言わせてくれなくちゃだワ
連載
ショートプロぱーてぃ/Z80's Bar/DōGA・CGA
X-BASIC調理実習/X68000マシン語プログラミング
● 新機種X68000SUPER-HD/EXPERTII/PROII
● ラジコンスティックの製作
LIVE in '90　バーニングフォース(OPMD対応)
THE SOFTOUCH　天下統一/ポピュラス/Hyperword
全機種共通システム　インタプリタ言語STACK

[第1話]

いろいろ選択の自由?

TAKAHARA HIDEKI 高原 秀己

金満大国ニッポンの巨大な富を狙って，海外から一流アーチスト，ミュージカル，コンサートが続々と来日するにぎやかな昨今。

マドンナ2度目の来日コンサート初日となった4月13日，そのことをまったく知らなかったぼくは，東京ドームで開催された「日米レスリングサミット」というプロレスのビッグマッチに出かけた。全米最大のプロレス団体といわれるWWFもまた日本の富を狙ってやってきたのだ。

日本初興行とあって，ハルク・ホーガン，ランディ・サベージなど主要レスラーが大挙来日，日本人レスラーとの対決も含めた超豪華メニューを並べてくれた。1万5千円也の異常に高額なチケット（前売りが始まってすぐに押さえた）に見合うか否かは別にして，とにかく面白かった。ホーガンの豪快で明るいファイト，天龍対サベージの熱戦など次々と存分に楽しめた。

UWF人気でわかるとおり，日本のプロレスファンはこのところ真剣勝負好きだ。徹底したショウアップが売り物のWWFは趣向がまったく逆。強い拒否反応を示す声もあったのだが，ここ20年ほどプロレス大好き人間を続けているぼくにはピンとくるものがあり，WWFに期待した。

この試合があったことによって，今後いろいろと波及効果が出てくるのは確実だ。ぼくとしては，「プロレス」のあり方がひとつ増えたことが最も大きな成果だったと思っている。

「職業選択の自由～アハハーン」というCMがあったが，選択肢が増えるということはすごく素敵なことだ。生活を確実に豊かにしてくれる。

ここ数年で日本は金満大国などと呼ばれ，みんなが金持ちになったようにいわれている。実際には土地価格の急騰や円高などで見かけ上の資産が増えただけのことであり，個人レベルで一斉に所得が増えたわけではない。ただ，ここ数年で明らかに変わったなと実感するのは，どんなことでも「選ぶ」という作業ができるようになってきたことだ。

典型的な事例をひとつ。一杯飲みに行ったときの光景。

「じゃあとりあえずビール。あ，どこのですか？」

「キリンですが」

「ほかのないですか？」

「(当然といった顔で)ウチはキリンです」

この極めて常識的なやりとりは，アサヒのスーパードライの大ヒットによって，ようやく打開された。

もちろんまだキリンしか置いていない店が圧倒的に多い。ただ，以前はキリン以外のビールがないかと確認する作業がすごくはばかられた。それを試みるだけで，すごいプレッシャーがあった。店員は聞かれること自体が非社会的行為であるかのような顔をし，一緒の席にいる人間たちも，

「どこのでもいいじゃないですか」

と諭したり，挙げ句の果てには，

「ビールは飲んでしまえばビールだ」

と落ち着いて考えればムチャクチャなことを平然とした顔で説明する人物までハバを利かせていたのである。

「すいません。キリンしか置いていないんです」

という台詞がごく自然に聞かれるようになった変化は大きい。

これ，たまたまスーパードライがヒットしたからで，選択肢うんぬんとは関係がない，と考える向きも，もちろんあろう。

だがそうだろうか。アサヒビールの新製品開発や宣伝広告努力もあっただろうし，スーパードライが力のある商品だったということは確かだろう。

しかしながら，世の中の風潮が以前ほど画一的であれば，いかにスーパードライに商品力があり，アサヒビールが頑張っても，普及には限界があったはずだ。むしろそちらに大きな要因があったように思う。スーパードライが出現していなくても，遅かれ早かれ別のヒーローが登場したはずだ。社会的風潮がキリン一辺倒では満足できなかった，と考えるのはとっても自然だ。

「政治は自民党，野球は巨人，自動車はトヨタ，ビールはキリン」――。

これが戦後から高度成長を遂げてきた日本の実態である。どうも日本人って人種は，不思議と好き嫌いを大勢に合わせていないと不安に感じる気質を持ちすぎていたようだ。「みんなが右なら自分は左」というなんでも反対論者にしても，大勢がどっちを向いているかを十分認識しているからできること，で結局は「大勢」派と同根だ。

ここにもうひとつの日本人の気質，「お上がそうするのだから」，「そうなっているのだから」，さらには「一番売れているのだから」という妥協主義が加わってしまうから，事態はさらに悪化してしまっていた。

自民党が昨年の参議院選挙でかつてない大敗をした。リクルート事件やマドンナ旋風があったものの，これなどは政治に対して自民党一辺倒ではないという気運が社会全体に満ちてきたひとつの証拠だ。

戦後40年以上たって，ようやく日本人がより大きな価値観を求めて，従来のパターンから脱却し，新しい素敵な選択を求めて動き出したのだろう。こうした指向性の変化に呼応するように，社会環境もテクノロジー武装や，円高と外圧による輸入拡大によって，ますます「選択」を許すように変わってきたことも見逃せない。

とにかくいろいろなジャンルで選択できる時代になったのは喜ばしい。

こう考えてふとパソコンを眺めると，あら不思議。PC-9801の勢いはまったくとどまる様子がない。シェアは依然として50%前後あるし，セイコーエプソンの互換機を含めると60%以上にまで達してしまう。もうちょっと視点を拡大してみると，インテル系のパソコンばっかりなのだ。Macintoshを別にすると，X68000だけがポツンと孤軍奮闘しているのが実に異様に感じられてしまう。

でもご安心を。WWFの来日もスーパードライの大ヒットも突然のことだった。98ユーザーだって，みんなが98を使うことなんかは望んでいないのだから。

第39回　知能機械概論――お茶目な計算機たち――

マックやめますか，それとも人間……

X68000に関してもその傾向は多分に見られるのですが，やはりなんといっても，Mac(Macintosh)は正真正銘の麻薬です。麻薬たる条件は何かといえば，まず，一度足を踏み入れると抜け出せない快楽でなければなりません。そして次に，心身にとって不都合なことがどんどん起きてくるということでしょう。Macは麻薬であることを本記事で示して，Mac中毒者になる人がひとりでも少なくなることを，中毒者の端くれとして祈っています。

マルチスクリーンの快楽

最近，Macにカラーボードを買ってきて，カラーディスプレイをつなげました。これが底なしに素晴らしいのです。単にモノクロがカラーになっただけではもちろんありません。面白いのは，もとのMacの本体のほうの画面も独立に生きていて(図1)，仮想的な画面領域をどのように2つのディスプレイで表示するか設定できるのです。

このようなマルチスクリーンの機能は最初からあるシステムソフトウェアの機能です。そして設定によって画面を縦に連結する（ディスプレイをMacの上に乗せる）ことも横に連結することも自由であることは，オペレーティングシステムの機能に含まれているわけです。少し暇そうな人が近づいてくると，僕はつい目の前で2つの画面にまたがってウィンドウを開き感心させてしまいます（ひんしゅくを買っているかもしれないが）。

症状1　自分のMacをなにかと自慢する

大きな画面ひとつでたくさんのウィンドウを開くというのがトレンドのようですが，このようなマルチスクリーンのほうが便利に感じられることもあります。たとえば，ひとつのウィンドウ上の仕事に集中しているときは，まわりのウィンドウに別の仕事が表示されているよりも，独立したディスプレイにそのウィンドウだけを開いて処理したほうが，気分的にスッキリするからです。

図1　マルチスクリーンの例

ディスプレイを本体の右に置く
OSでレイアウト設定

ディスプレイを本体の上に置く
OSでレイアウト設定

マルチファインダの快楽

僕のMacはメインメモリは5Mバイトで，マルチファインダを走らせています。マルチファインダは同時に複数のプログラムを（手で）切り替えて実行することができるシステムです。UNIXのように完全なマルチタスクではないのに，十分だと感じられるのはなぜかといえば，ウィンドウどうしがUNIXに比べて密接に結びついているということがあるかもしれません。

UNIXでマルチウィンドウといっても，普通に開くだけでは，ただ端末が増えてマウスなどでテキストの若干の移動ができる程度で，各ウィンドウで独立に処理を進めるということが基本でしょう（かなり主観的だな），が，Macの場合はシステム全体として統合化の思想がありますので，各ウィンドウの独立性はそれほど必要ではないということもあるかもしれません。

マルチファインダだと，たとえば，日本語ワープロで文章を編集し，その中に図を貼り込むときに快適です。リンク指定などというものをやると，日本語ワープロと図形作成プログラムが心地よくリンクしてくれます。図形作成プログラムでちょっと貼り込んだ図を修正すると，文章の中の同じ図もそれに連動して更新されるのです。

ハイパーカードの快楽

ハイパーカードのとりこになった人の表現を借りると，「オブジェクト指向の概念をカード型データベースの中にマルチメディアを取り込んで究極的な形で実現したもの」なのだそうです。とにかく自分で自由にデータベースを作ることができて，あれやこれやとワクワクしながら使えるのです。

研究室の自分の机の上にMacは置いているので，とりあえず作り始めたのも，研究メモと参考文献データベースが混じったような感じのものです。まず表紙のスタック（カード）があります。そして，研究テーマの名前なども書いてあり，そこをマウスでクリックするとそのテーマの最初のスタックに行きます。そのスタック上では，指定されたフィールドに文章を書いてもいいし，直接お絵描きツールで書き込むこともできます（大事な部分は鉛筆ツールでグルっとマルをします)。そして，参考文献があるものは横にボタンをつけてあります。そこをクリックするとその文献データの入ったスタックに移動します。参考文献のスタックのほうは著者名や年度などフォーマットがかなり固定していますが，いくらでも文章を付け加えることもできます。

このスタックでは，自由にものをしゃべらせることもできるのです。ただし，英語でないときちんとしゃべりません。いまは，研究室の事務の人に電話がかかったときのためのボタンが1個つけてあるだけです。そのボタンを押すと，「サカイサン，デンワダーヨー！」とガイジンさんのようにしゃべります。

ハイパーカードを使いはじめるとなんだか賢くなったような気がしてきます。人はいいことを思いついても，忘れてしまってはゼロです。しかし，このようなツールを使えば，アイデアを書き残せるだけでなく，いろいろ別のこととの関連も残したまま記録することができます。生きた知識，つまり頭脳の延長としてのマシンに近くなっているような気がします。

症状2　素晴らしいことにMacを使っていると思い込む

少しずつ痛い目にあう

Macはいくらでもお金を食うとよく言われますがこれは真実です。最近は日本で買う値段もだいぶ安くなってきたとはいえ，それでもそんじょそこらではありません。待ちに待ったMacポータブルがあのような値段でしたし，ついにUNIX(X-Window)とMacのオペレーティングシステムが真に融合したといわれる40MHz68030マシンMacIIfxに至っては，UNIX抜きで本体（ハードディスク80Mバイト）で1,678,000円と聞いては，ちょっとした大衆車の値段とい

えるのでしょう，よく知りませんが。それでも中毒者は買うのでしょう，ああ恐ろしい。人ごとではないか？

Macはお金を食うだけでなく，時間だっていくらでも食います。とにかくなにかいじっているだけで楽しいのです。PC-9801に向かってMS-DOSコマンドを入力して「へへへ」と笑っていたら逃げ出したくなりますが，Macに向かってにやついていてもごく自然です。

症状3 Macに都合のよいことを言い出す

ウイルスにやられる

ついに，ウイルスが僕のMacにも侵入しようとしたのです。しかし，なんとかそれは，ウイルス侵入防止プログラム（GateKeeper）によって，ぎりぎりのところで防止することができました。ウイルスは知り合いの人から借りたディスケットの中に潜んでやってきました。そのディスケットを差し込んだとたんに，画面に「ファインダが無理やり書き換えられようとしている」とのメッセージが出て，なんとか大事に至らなくてすんだのでした。

さっそく，ディスケットの持ち主にそのことを言いました。そして，いろいろ，持ち主が原因を調べたところ，別のウイルス発見プログラム（Disinfectant）によって，ハードディスク上のあるプログラムが感染しており，それが原因だったということがわかったようです。朝日新聞はときどき大袈裟に記事を書きますが，このような話はきわめてよく聞く話なのです（自分の目の前に初めて現れたときはさすがにびっくりしましたが）。

Macがウイルスに感染しやすい理由のひとつは，Mac自体のシステムにあるのでしょうが，もうひとつはアマチュアの作ったプログラムが大量に流れているという環境にあります。ネットワーク（junet）を通じてMacのプログラムが毎日毎日届けられるので，面白そうなものを僕もダウンロードしています。まあ，こうやって手に入れるものは，かなり選択されて残ったものでしょうから，それほど心配はないのかもしれませんが，いろいろなところに得体の知れないソフトはころがっています。でも，今回未然に防御したGateKeeperにせよ，Disinfectantにせよ，それらも，そのような無料で配布されるプログラムのひとつなのですから，変といえば変な話ですね。

爆弾にやられる

Macをいじっていると文字どおり爆発に巻き込まれて痛い目にあいます。システムエラーが起きて復旧できなくなると，Macのオペレーティングシステムは，爆弾の絵が書かれたメッセージを出して果てるのです。この爆弾マークは，案外と頻繁に現れます。とくにマルチファインダで多くのプログラムを同時に動かしているときなどは，この爆弾に見舞われると傷は浅くはありません。図を作りかけていたときや，文章を書きかけていたとき，何度「セーブしておけばよかった」と思ったことでしょう。

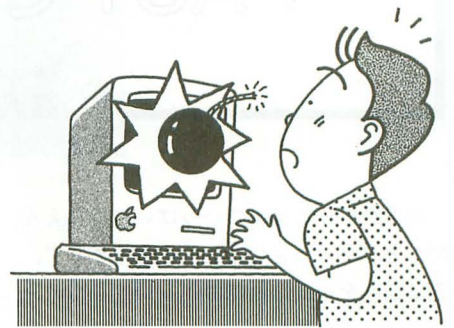
Macでは，システムフォルダというディレクトリに，ある特定の種類のファイル（init, cdev型）を入れておくだけで，システムの拡張やパッチができます。そして，数多くの素晴らしいそのような種類のプログラムによって，思いもかけないようなことが実現されてきました。しかし，それがこのような爆弾の火種になっているのです。相性の悪いものをシステムフォルダに入れておくだけで，最悪のときはMacが立ち上がらなくなるときさえあります。それも，自分なりに手間暇かけて愛着も深まると，ちょうど頃合を見計らって起こるのです。

症状4 何度痛い目にあってもめげない

最近のMac用の日本語ワープロでは，そのようなMacの状況も考えてか，何分おきとかに，自動的にセーブをしてくれるような機能を持ったものも出てきています。でも，「あー，めちゃめちゃになった，もう一回編集やり直しっ！」というようなときに，元の文書はすでになくなっていたら困ります。「余計なお世話」ですね。ちなみにこの日本語ワープロの場合，このワープロ自体がよく暴走することがあるために，このような機能を付けているという噂もあります。

これも書かねば

思いつくままにいろいろ書いてきましたが，本当の問題は実は次の2つであると思われます。

1．Macの閉じた世界

2．アメリカ生まれ

最初の問題は，かなりMacに浸ってきて，逆にこれだけではもの足りなくなったときに，ふと感じるものです。そう簡単に解決できるような問題ではありません。ひとつの革新的なアプローチがMacのもつインタフェイスであり，これを打ち破るような枠組みはまだ明確には存在していないと考えられるからです。ただ，最新機種であるMacIIfxは個人で購入するマシンではなくなってしまったとはいえ，この問題に対してはひとつの答えを出したといえるでしょう。

2つ目の問題は，別にアメリカ産だからまずいというのではもちろんありません。ただ，日本で日本語を用いて使うという場合に，どうしても限界が見えてしまうのです。日本語ワープロには，英文ワープロと違い，変換を始めとして日本語特有の処理がいろいろあります。したがって，Macの特長である「プリンタで印刷するイメージがそのまま編集中に画面でも見られる」という思想に忠実であるとすると，処理が重くなってしまい，遅くて話にならないワープロが誕生してしまうのです。

さらに言うならば，Macを日本で売っている代理店の問題もあります。Macをどのようなマシンとしてとらえているのかということを始めとして，不満はいくらでもありますが，最近は大学生協との提携なども始めたようで，もしかしたら少しは変わってきたのかもしれません。というよりは，変えざるをえなくなってきたのでしょう。

最近，小学校頃の文集が出てきて僕の性格について友達が次のようなひとことを書いていました。

「一緒に遊ぶと気が合うし，おもしろい。ただ，よく自慢をする」

うーむ。では，また。

第48回

猫とコンピュータ
FASTでなくちゃ

Takazawa Kyoko
高沢 恭子

今回はホンニャアにはちょっとお休みしてもらって，マジメにファーストフードについて考えてみたいと思います。早いという長所のウラには，成人病につながる高脂肪，高カロリーというワナが……。

両手を広げて優しさいっぱいにほほえんだかと思うと，急に冷たく心をとざして，春は悩みを抱えた地球のように，立ち止まっては，また歩む。

気まぐれなお天気のイジワルで，恒例のFBIのお花見が，初の流会になった。花は満開，空は青いのに，前夜の雨で地面が濡れていたのだ。

桜の花なら，ことしもあちこちでたくさん観賞できた。旅先のホテルのラウンジの窓をおおいつくしたみごとな眺め。車窓をよぎっていった，幻想のようないくつもの一群。新宿の家の門扉のそばで40年咲きつづけるソメイヨシノ。それからわが家の向かいの保育園の庭の1本。テレビの画面では，実物の何倍もの大きさで1輪ずつ見せてくれる。

でも桜をほんとうにたたえるなら，根をはる地面に自分も腰をおろして，花が生き生きと咲いている時間をしばらくいっしょに過ごしてやるのがよいと思う。だからお花見には地面がとってもたいせつなのだ。

「咲く」「うららか」が桜の語源だと，専門家のお話にあった。桜の息づかいを幹の間近で感じながら，時おり花びらを手に受けられたら，それは最もうららかな春の味わい方だ。

恐怖のセットメニュー

スーパーマーケット前の広場でも，春はうららかだ。花壇の三色すみれやスイトピーが，日ざしに包まれ，微風にそよいでいる。

午後3時をまわって，広場に面したガラスばりのハンバーガーショップも大にぎわい。高校生も，これから塾に出かけるらしい小学生もいる。どこにでもあるファーストフードの光景だ。小さな子供をつれたお母さんも多い。そう，お母さんがおやつをつくるのでなく，つくってあるおやつを子供といっしょに食べるお店なのだ。

「ファーストフード」は注文にすばやく応じられるとか，調理ずみですぐ食べられるといった意味らしい。アメリカから来たハンバーガーやドーナッツのショップが筆頭になるけれど，列車の駅にある立ち食いそばや牛どんのお店だって仲間なのだ。

1人ひとりが仕事もスケジュールもパーソナルな時代だから，調理してある食品や，たやすく食べられるものがたくさん売られるようになった。でも立ち食いしてもファッションになるのは，なんといってもハンバーガーショップだ。それが気づいてみると，家の周囲だけでも7，8店はある。

とうとう中学校の最高学年をむかえたトオルのスケジュールも，以前に増してめまぐるしく複雑になってきた。ブラスバンド部はあいかわらずで，じつはあんなに深く考えて見切りをつけるはずだった生徒会で副会長をつとめている。会長はマドンナブームの影響か，女の子のツボイミサコさんだ。いくらなんでも受験の年だし，早めに後輩にバトンタッチをして，自分の態勢もととのえてほしい。こちらはそう思っても，いつも何かの計画表を片手に，学校と家を日に何度も往復している。

家に飛びこんできたときに，ちょっと時間にゆとりがあると，ホッとした気分から何かおやつをということがよくある。1日の栄養プランから考えて，飛び入りの食品の内容にはもう少し注意をはらわなくてはと思いながら，175cm，60kgの育ちざかりに圧倒されて，つい近くのファーストフード店で仕入れてしまうことが多いのだ。

学校からの「給食だより」で，飽食による成人病の若年化をいましめて，「ファーストフードなどの普及で，かんたんに高カロリーをとってしまう」と指摘されなくても，だいぶ前から，母親として独自のおやつを作る努力をしていないことを，おおいに反省はしていた。

ことしの2月ころ，朝日新聞に国民生活センター調べというハンバーガーについての記事が載ったことがあった。

「テストの結果によると，ハンバーガーの栄養面の大きな特色はカロリーが高いこと。1個あたりの重さはおむすびとほぼ同じだが，カロリーはハンバーガーのほうが2倍もあった」「脂肪とは対照的に乏しかったのがビタミンAやビタミンCなど」「ドリンク類では，シェイクが最も高カロリーで，ハンバーガー1個に相当」「栄養のアンバランスはセットメニューによってさらに拡大しがちだ。18セットについて調べたが，全体的に高脂肪，高カロリー。最も高カロリーだったのはテリヤキバーガー，シェイク，フライドポテトのセットで1580カロリー」。などという内容だった。

バーガー哲学

ちょっと背すじの寒くなる思いで新聞記事を見ていると，トオルが笑いながら，「お母さん，こんどはこれ読んでみて，びっくりするから」と，雑誌の付録の薄い冊子を持ってきた。『特集 今日の我々の生活はファーストフードによって支えられている!!』。

ハンバーガーショップの大特集だ。あるある。ウェンディーズ，ケンタッキー，ドムドム，ファースト・キッチン，マクドナルド，モスバーガー，ロッテリア。各店の誕生や由来，特徴，ベスト人気メニュー，値段やカロリーの一覧表。裏話やものしり事典，ついでに各チェーン店の全国分布図から年間総生産量まで，すべてリサーチさ

れている。知識を増やしてじょうずにメニューを選ぼうというわけだ。

アメリカで誕生したのは50年ほど前で，日本に上陸してから約20年，国産のチェーン店も生まれて，すっかり日本人の生活や風土に定着したファーストフードの店。とくに若い人の好みに合う味とスピーディな点で，個人単位の食生活を支えてしまうのはほんとうだ。この日は偶然に，ファーストフードの批判と支持の両方を見た。

クレープ，たこ焼き，ソフトクリーム，焼きそば。スナック類をつくりながら売る店は，まだまだいくらでもある。そのなかでメニューもバラエティに富んだハンバーガーショップの人気は高く，業界1位のマクドナルドはことし2月にモスクワに出店して話題になった。気軽さ，求めやすさが自由の象徴となって，ソ連に上陸したという感じだ。

こういう店にみんながあつまる理由は，なんといってもその速さと会計の明快さだと思う。まず，ほしいものをすぐ味わうことの満足。味のよしあしより時間の効力がものをいう。同じものを1時間後に食べてみれば，それは実感できる。

それから，支払いの安心感。いつか新聞の投書に，ある食堂でビールを注文したら枝豆がついてきて，それも計算書のなかに加えられていたのがなっとくできないという言い分があった。カウンターで自分が指定した品目だけの金額を事前に支払うやり方は，その点とても明朗だ。

昔からあるおすし屋さんも，目の前で注文のすしダネをにぎってもらえるのだから，かたちはfastなのだけれど，食べおわるまで会計は不明だ。これは店のふんいきや板前さんの腕前まで売り物なのだから，成り立ちがまったくちがうのだろう。ほんとのファーストフードは，均質な品をお金と交換するだけのようだ。ことにマクドナルドなどは，誰がつくってもまったく同じものができるための，共通の機械と処方せんを使っているのだそうだ。

均質という点では，従業員の接客態度もマニュアルが徹底していて機械的だ。でもみんな元気で明るい。同じ接客マニュアルでも，「5つのNO」なんていうのをかかげている電器販売店が，なるべく機械的販売に近づけようとしながら，機械に劣る接客をしているのとくらべたら，ずっと気持ちがいい。繁盛してあたりまえだ。

軽子坂のRUKIT

ほしいものをすばやく手に入れたい。望んだものを的確にもとめたい。

パソコン通信の利用がさらに普及して，いろいろな情報のデータベースを活用したいという人が増えているそうだ。

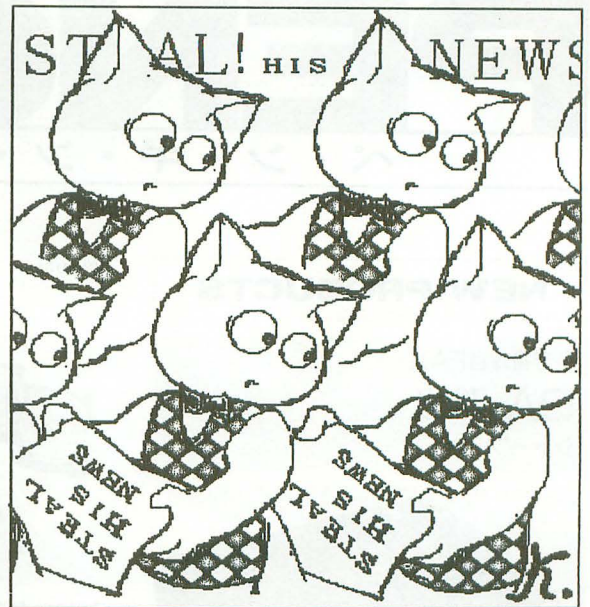
いまはふつうに生活していても情報は限りなく流れこんでくるが，逆に自分が求めている情報を見のがしてしまうことも多い。そんななかで，必要な項目や用件について，なるべくたくさんの情報をさがしあつめるのがデータベースの利用だ。

情報がすべて。情報がつぎの情報を生む共食いみたいな時代に，ほしいものだけ引き出して役にたてようというのだ。パソコンがいよいよ力量を見せてきた。

飯田橋の軽子坂MNビルは話題のインテリジェントビルで，とくに2階の「情報図書館RUKIT」が注目されてきた。8種類22台のマシンによって，誰でも国内外の20あまりのデータベースを利用できるのだ。桜が7分咲きの，ちょっと風のある日，夫の会員登録につきあって出かけてみた。

外観もインテリアもまさしくインテリジェントビル，ひんやりと金属的でムダがいっさい見あたらない。2階の図書館では，整然と並んだマシンの前で，2人の男性がサーチャーの資格を持つ館員のアドバイスを受けながら検索をしているところだった。

わが家のマシンルームからでも，費用と時間を無視すれば，目的の情報はかなりのところまであつめることができる。加入しているいくつかの商業ネットの中の，大きなデータベースから，項目やキーワードを自分なりに絞って検索していくのだ。夫はじっさいに，ビジネスの目的でたびたび利用している。

しかし，本格的に専門の分野などで漏れのない情報をあつめようとしたら，たくさんのデータベースを対象に，あらゆる角度から選び出さなくては満足できない。それに，この情報図書館を利用すると，個人で個々のデータベースを利用するより，費用がグンと割安になるそうだ。

会員の登録をすると，カード制でキャッシュレスになるが，このカードは銀行からの支払いだけでなく，ビル内の設備のキー機能も持っているところが画期的なのだそうだ。日本で初のインテリジェントカードということで，テナントや会議室の鍵，喫茶ルームのオーダーにも使える。

まだ大繁盛というほどのようすはないものの，情報の海から，「マックフライドポテト，ひとつ」なんて感じで，情報の迅速(FAST)提供をしてもらえる日が，だんだん近づいているらしい。

おやつの愁い

出た。アメリカの大新聞をはじめ15紙にマクドナルドを名指しで批判した市民団体の意見広告が。全米心臓病救助協会というところの発言で，先にこちらの国民生活センターが調査した結果の発表を，さらに過激にしたようなものだ。脂肪が極度に多く心臓にたいへん悪いというデータつきの意見で，「毒を盛られるアメリカ」の見出しは迫力がある。

ファーストフードの食事は空腹を満たすのが第一だから，質については誰もはじめから期待していないかもしれない。ハンバーガーでなくっても，ちょっと手を出してしまうおやつはファーストフードだ。みんな気をつけないと，食べ物が多すぎる。

PENGUIN INFORMATION CORNER

ペ・ン・ギ・ン・情・報・コ・ー・ナ・ー

NEW PRODUCTS

留守番録音FAX
DA-7000
シャープ

DA-7000

シャープは，留守番機能を持ったG3FAX「DA-7000」(178,000円）を発売する。留守番電話機とFAXを一体化したもので，留守中にかかってきた電話の音声だけでなく，送られてきたFAXの情報もテープに記録することができる。V.I.M.S.(Voice＆Image Memory System)という新機能を搭載することにより，電話の音声情報とFAXのイメージ情報を同時にテープに記録することが可能になったもの（電話とFAXの切り換えは自動的に行う)。これにより外出先の電話から自宅の留守番電話の伝言を聞くように，外出先の一般FAXからリモート操作して自宅に送られてきたFAX文書をプリントアウトすることができる。リモート操作では文書以外に，相手先FAX番号・着信枚数・着信時間などの着信リストも取り出せる。また転送に設定することにより，FAXがテープに記録された時点でポケットベルを呼び出してくれる。FAX受信中に紙切れになった場合はテープに代行受信される。60分テープでA4標準用紙が最大100枚記録できる。

〈問い合わせ先〉
シャープ㈱　☎06(621)1221，03(260)1161

携帯用CD-ROMシステム
KV-M101
松下電器産業

松下電器産業は，液晶ディスプレイ付きの携帯型CD-ROMビューアシステム「KV-M101」を発売する。大型液晶ディスプレイとCD-ROMドライブを一体化したもので，外でもCD-ROMのデータを見ることができる。16ビットCPUを搭載し，CD-ROMドライブ以外に3.5インチFDDも内蔵，セントロニクス準拠のプリンタインタフェイスも持っている。表示は640×400ドットのバックライト付きLCDで行う。読むことができるものは紀伊國屋書店・日外アソシエーツのデータベース「バイブルスシリーズ」や日本法律情報センターの法律判例情報「リーガルベース」などのCD-ROMディスク。またデータは検索・表示を行うのみではなく，付属の日本語編集機能でワープロのように編集も行える。さらにオーディオ機能も持っておりCDの再生も行える（内部スピーカーまたはイヤホン端子で聴く)。価格は未定であるが従来の同社CD-ROMシステム(約63万円)の半額程度になるもよう。

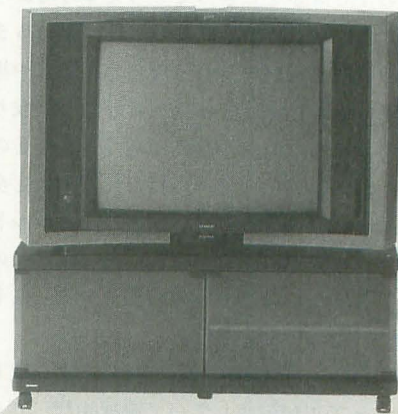
KV-M101
〈問い合わせ先〉
松下電器産業㈱　☎06(908)1121

世界最小のビデオコンポ
GET'S
日本ビクター

GET'S

日本ビクターは，世界最小・軽量のVHSビデオムービーコンポ「GET'S」を発売する。システムはS-VHS-Cビデオデッキ，CCDビデオカメラ，液晶モニタ，テレビチューナーなどのユニットで構成される。S-VHS-Cビデオデッキは，新開発の薄型軽量LTメカの採用により世界最小サイズ（530g)を実現，音質は高音質VHSステレオHi-Fiを搭載した。CCDビデオカメラも世界最小サイズ(160g)で，36万画素の解像度のCCD搭載，ピントはパンフォーカス，最低照度7ルクスとなっている。液晶モニタは，小型スピーカ内蔵のバックライト付き3インチカラーLCDモニタ（VTRリモコンあり)。テレビチューナーも小型設計(165g)で16チャンネルプリセット方式（ロッドアンテナ内蔵)。ほかにバッテリーパックとACアダプタなどが付属する。価格はフルシステムで288,000円であるが，希望ユニットのみのシステム構成も可能である。

〈問い合わせ先〉
日本ビクター㈱　☎03(5684)9311

ホームシアターテレビ
31C-E100
シャープ

31C-E100
シャープは，家庭で映画などを楽しむため映像・音声の両方に新技術を採用した31型ホームシアターテレビ「31C-E100」(275,000円）を発売した。画質面では，原色を忠

実に再現するため赤緑青の3原色に黒を加えた4原色を採用。音質面ではヤマハ㈱と共同開発した30Hzのアクティブサーボテクノロジーによる重低音再生の高音質化，センタースピーカとドルビープロロジックシステム，サウンドウイングの採用などが挙げられる。また，映画（アクション，シリアス，ロマンス，ホラー）や音楽（ライブ，クラシック），TV番組などソフトに合わせ最適な画質・音質がワンタッチで選べるAVメニュー&AVイコライジング機能も搭載。
〈問い合わせ先〉
シャープ㈱　☎06(621)1221，03(260)1161

自動的にCMカット
HV-S95/F93/F92/D91
三菱電機

HV-S95/F93/F92/D91

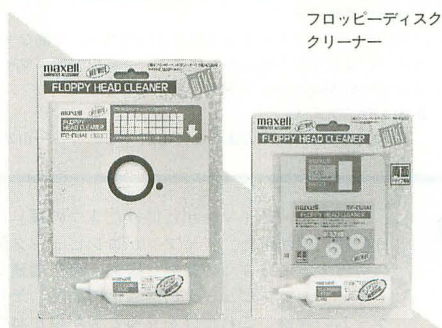
三菱電機は，自動的にCMをカットして録画することができるVTR「HV-S95/F93/F92/D91」(148,000円/95,000円/85,000円/66,000円)を発売した。2カ国語放送や音声多重の番組を録画しているときに入ってくるステレオやモノラルのCMをカットすることができる。CMのカットは音声多重番組の周波数とCMの周波数が異なることを利用して録画を一時停止することで行う。よって通常のドラマ・歌番組ではCMカットはできないし，CM自体が2カ国語放送の場合もCMのカットはできない。CMのカット録画を行えるものはTV欄に二②多副などの文字がついている番組のみである。
〈問い合わせ先〉
三菱電機㈱　☎03(218)4111

理系学生のためのポケコン
PC-1490U
シャープ

シャープは，全国大学生活協同組合連合会（大学生協連）と共同で開発した，ソフト内蔵型ポケコン「PC-1490U」を発売した

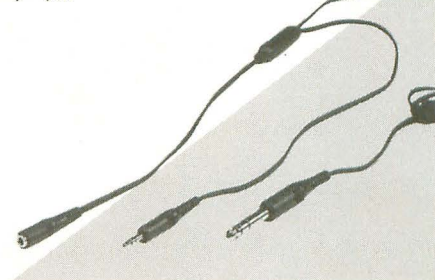
PC-1490U

(組合員価格19,800円)。大学生協連が組合員の学生・教職員から集めた22本のソフトからなるライブラリ「電言板」を組み込んでいる。ソフトは，理系学生のSTUDY用として「n進数変換」，「複素数計算」，「トランジスタの静特性」，「分子量計算」，「シンプソンの公式による積分計算」など9つ。LIFE用として「レコードの時間計算」，「天中殺調べと相性診断」，「授業時間割」など7つ。TECHNICとして「ドットパターン作成プログラム」と「通信用サンプルプログラム」。ゲームとして「POCKET INVADER」，「ビンとりゲーム」など4つ。また，ポケコンPC-1490U自体は，シャープのポケコンPC-Uシリーズの新製品(7機種目)で，8ビットCPUを内蔵，32KバイトのRAMを標準装備（別売のRAMカードにより最大96Kバイトまで拡張可能）している。プログラム言語にはBASICを使う。
〈問い合わせ先〉
シャープ㈱　☎06(621)1221，03(260)1161

アクセサリー新製品
Wikiシリーズ
日立マクセル

フロッピーディスククリーナー

日立マクセルは，コンピュータ関連アクセサリーの「Wikiシリーズ」に6種類の新製品を発売した。シリーズ2回目として今回追加されたものは，1500ガウスの防磁力で大切なデータを磁気から守る「5インチフロッピーディスク防磁ケース」(1,400円)と「3.5インチフロッピーディスク防磁ケース」(1,000円)，情報の分類・整理に便利な「5インチフロッピー用インデックスラベル」(300円)と「3.5インチフロッピー用インデックスラベル」(300円)，クリーニング効果が高いウエット&ワイプ方式の「5インチ用フロッピーディスククリーナー」(2,500円)と「3.5インチ用フロッピーディスククリーナー」(2,500円)の6製品。
〈問い合わせ先〉
日立マクセル㈱　☎03(241)9736

X68000の音をMIDIとミックス
オーディオ・ミキシング・ケーブル
電波新聞社

オーディオ・ミキシング・ケーブル

電波新聞社は，MIDI音源とX68000の内蔵音源の出力をミックスして音を出すことができるケーブル「オーディオ・ミキシング・ケーブル」(1,300円)を発売した。ステレオミニプラグとステレオ標準プラグの2系統をステレオミニジャック1系統にまとめて出力するものである。
〈問い合わせ先〉
電波新聞社　☎03(445)6111

ゲームソフトBGM集発売
ゲームボーイ　グラフィティ
日本コロムビア

日本コロムビアは，ゲームソフトBGMなどを集めたCD/カセット「ゲームボーイグラフィティ」を発売する。収録曲は，「テトリス」，「スーパーマリオランド」，「ソーラーストライカー」など任天堂のゲームボーイのゲームソフト8つから選んだものと，各ゲームのオリジナルBGMを収録したものなど16曲。なお特典としてキャラクターシールが1枚封入。税込み価格はCDが2,600円，カセットが2,300円となっている。
〈問い合わせ先〉
日本コロムビア㈱　☎03(584)8236

FILES Oh!X

このインデックスは，タイトル，注記――筆者名，誌名，月号，ページで構成されています。ゴールデンウィークも終わり，もうすぐうっとうしい梅雨がやってきます。ついでに中間試験なんかも……。

一般

▶輝け！　春のパソコンガイド
新入学のシーズンにあわせて，各メーカーの最新機種を大紹介。PC-98/286シリーズ，J-3100シリーズ，タウンズをはじめ，X68000もPRO II，EXPERT II，SUPERの発売とあわせて解説されている。MacintoshやAmigaも紹介。――編集部，LOGIN，8号，114-127pp.

▶ネットワーカー・ホリック第19回
プロの手によるオンラインマガジン「KENNEDY」を紹介。PDSはX68000用ポリゴンフライトシミュレータ「ポリゴンくん」，PC-98用ロールプレイングゲーム「舞瑠華」。今年度からNTTが設置するデジタル公衆電話で，アウトドアでもパソコン通信が楽しめるようになるという話題など。――編集部，LOGIN，8号，198-199pp.

▶ハイテク地獄耳
女性向けにローズレッド色調のデザインを採用したシャープの電子手帳「PA-7550」と，新しい電子手帳用ICカードを紹介。――編集部，POPCOM，5月号，131p.

▶ DōGAアマチュアCGアニメーションコンテスト
2月25日エルムホールで行われた第2回アマチュアCGアニメーションコンテストの表彰式の模様をレポートする。――編集部，ASCII，5月号，393p.

▶'90 AOUショー
メッセで行われたAOUショーの模様をレポートする。――MUNEP，マイコン，5月号，209p.

MZ-80B/2000/2500/2800

MZ-2500（MZ-5Z001BASIC）

▶ PAKETY
パケティを3匹救助して，最下階にある穴に落ちると脱出成功。迷路探索ゲーム。――あむーる，マイコンBASIC Magazine，5月号，127-128pp.

MZ-2500（M25-BASIC）

▶ RIDO
時間内に4色のパネルをすべて消すパズル。――蒲生敬，マイコンBASIC Magazine，5月号，129-130pp.

X1/turbo/Z

X1シリーズ

▶誌上公開質問状
X1にディスクドライブを接続したいのだが，可能なドライブにはどんなものがあるか？――多田太郎，マイコンBASIC Magazine，5月号，90p.

▶ SOLID GOLF X1
3Dゴルフゲーム。木の上からでも打てる？――山本弘，マイコンBASIC Magazine，5月号，157-158pp.

▶ BRAINS
一筆描きゲーム。面データはパソコンが勝手に作ってくれる。――HARIKUN-TM，マイコンBASIC Magazine，5月号，159-160pp.

▶ LEMMY X1
1990年2月号のMSX2用同名ソフトの移植版。炎に焼かれる前にレミングたちを全員海に飛びこませて溺死させるというゲーム。――小國健，マイコン，5月号，214-220pp.

▶ LET'S PROGRAM
今月の宿題発表は「テスト得点の修正」。共通一次でもあった「かさ上げ修正プログラム」のサンプルのHu-BASIC版が掲載されている。――藤本健，マイコン，5月号，229-237pp.

X1＋FM音源ボード（要FM音源ドライバ）

▶ BURNING FORCE ――GRASS LAND――
ナムコのゲームミュージックプログラム。――CoM，マイコンBASIC Magazine，5月号，189-190pp.

X1turboシリーズ

▶ NEW SOFT
セレクテッドソーサリアン3を紹介。――編集部，LOGIN，8月号，14-15pp.

▶月刊ソーサリアンニュース
ソーサリアンのシナリオ「セレクテッドソーサリアン3」の前半パートを紹介。――編集部，テクノポリス，5月号，30-31pp.

▶ ESCAPE
ホバーシップを操作して一定時間内に坑員を回収し，出口へ脱出する。迷路脱出ゲーム。――内田真裕，マイコンBASIC Magazine，5月号，161-163pp.

X68000

▶ NEW SOFT
新着ゲームのプロディア，グラナダを紹介。――編集部，LOGIN，7号，16-18pp.

▶ X68000新聞
新着ゲームのポピュラス，ラグーン，銀河英雄伝説パワーアップ＆シナリオ集，プロディアを紹介。PDSはフロッピー，ハードディスク，メモリエディット機能まで付いた高機能ディスクエディタ「DEDIT.X」。そのほかDōGA・CGコンテストの応募作品，サードパーティのアイ・オー・データ機器から発売された拡張メモリボードの紹介など。――編集部，LOGIN，7号，148-151pp.

▶最新ゲーム徹底解剖!!
英国生まれの人気シミュレーションゲーム，ポピュラスを徹底攻略。――編集部，LOGIN，7号，182-185pp.

▶ Software Review

参考文献

I/O　工学社
ASCII　アスキー
コンプティーク　角川書店
テクノポリス　徳間書店
POPCOM　小学館
マイコン　電波新聞社
マイコンBASIC Magazine　電波新聞社
LOGIN　アスキー

新刊書案内

「フォン・ノイマンが仕掛けた3つの罠」という副題に惹かれた。ノイマンがコンピュータ関係の分野で行った3つの重要な研究。自己増殖ソフトウェア，コンピュータの信頼性の問題，そしてゲームの理論。これだけを見ると何が罠かわからないだろうからくだけていおう。現在のコンピュータ化社会の問題点はすべてコンピュータ出生時から抱えていた問題であり，ノイマンはそのことに気づいていた，ということだ。現代社会は，何十年も前からわかっていた問題に対し何の解決も見出さないままコンピュータに依存してきたため，抜き差しならなくなっている。自己増殖ソフトウェアはコンピュータウイルスという形で現れ，信頼性の問題はコンピュータの構造からバグのないプログラムはまずあり得ないことを示し，ゲームの理論は現在隆盛を誇っている金融ゲームにあてはまる。ノイマン型のコンピュータは世の中を任せるには構造的な問題を抱えすぎている。本書はそれをわかりやすく語った本だ。一見センセーショナルに過ぎるが，冷静に考えてみよう。コンピュータはもともと軍事機器だったのだ。（K）

コンピューター社会が崩壊する日　逢沢明著　光文社
☎03(942)2241　新書判　235ページ　750円

書籍の価格は消費税込みです

ゲーム史に残る名作，ダンジョンマスターをエンターテイメントの見地から解説。――高橋ピョン太，LOGIN，7号，212-213pp.

▶ NEW SOFT

4月発売予定のアクションゲーム「グラナダ」，シンキングアクションゲーム「スライミャー」，追加シナリオセットの「大海令―限定版―」「南海の死闘―限定版―」「A列車で行こうII―限定版―」を紹介。――編集部，LOGIN，8号，18-21pp.

▶ X68000新聞

ニューハード速報。遂に発売されたX68000の新タイプ「PRO II」「EXPERT II」と，6月発売予定の「SUPER」シリーズを大紹介。周辺機器は光磁気ディスクユニット。付属する本格的ウィンドウシステム「SX-WINDOW」も解説。PDSはデータオンメモリで録画・再生を行うアニメーションソフト「ESV」を紹介している。新着ゲームは「ファーサイドムーン地球防衛軍II」「A列車で行こうII―限定版―」「大海令―限定版―」「南海の死闘―限定版―」「クォース」「LIFRAIM」。――編集部，LOGIN，8号，130-137pp.

▶新作ゲーム徹底解剖!!

ワンダラーズ・フロム・イース，ブロディア，キューブランナー，ポピュラスの攻略テクニックを解説。――編集部，LOGIN，8号，164-167pp.

▶ Software Review

新着アクション「サンダーブレード」「クォース」を紹介。――花ポン・X68000新聞社，LOGIN，8号，194-197pp.

▶特集春のゲーム万博

この春注目のソフト「ポピュラス」の攻略法を紹介。――編集部，コンプティーク，5月号，90-93pp.

▶ X68000SPIRITS

ダウンタウン熱血物語，グラナダ，クォース，サーク，グランディフロラム，闇の血族，天九牌の紹介。――編集部，コンプティーク，5月号，194-197pp.

▶ HARD PACK PAGE

X68000新シリーズ登場のニュース。「SUPER」「EXPERT II」「PRO II」シリーズの紹介と，SX-WINDOWの解説など。――編集部，コンプティーク，5月号，215p.

▶先取りおすすめゲーム

5月発売予定のアクションゲーム，「ブレード・オブ・ザ・グレート・エレメンツ」を紹介。――編集部，テクノポリス，5月号，12-13pp.

▶ GAMING WORLD

X68000オンリーのアクション「グラナダ」と，「バブルボブル」「キューブランナー」の紹介。――編集部，テクノポリス，5月号，16-19pp.

▶ SOFT RADAR

パズルゲーム「ブロディア」を紹介。――編集部，POPCOM，5月号，20p.

▶ゲームがオレを呼んでいる！

ダンジョンマスターのレベル7・8にアタック！――ボンセ松崎，POPCOM，5月号，74-75pp.

▶ WE ARE THE X68000 WORLD

80MバイトのHDを積んだ最新機種，SUPER-HDやEXPERT II，PRO IIを紹介。また近々発売が予定されている周辺機器，光磁気ディスクユニットやSCSIボード，ビデオ出力ボード，音声多重デコーダ内蔵のディスプレイテレビなども紹介している。SX-WINDOWについても簡単に解説。新着ゲームソフトは「LIFRAIM」「天下統一」「サーク」「銀河英雄伝説パワーアップ＆シナリオ集」。そのほか大量の文書の整理に使えるアイデアプロセッサ「Hyper Word」。――編集部，POPCOM，5月号，84-87pp.

▶誌上公開質問状

オリジナルのAD PCMデータをBeep音に使うにはどうするか，X1turbo IIとX68000共通に使えるディスプレイにはどんなものがあるか，PROシリーズには拡張I/Oボックスは使えるのか？　などの質問に答えている。――多田太郎，マイコンBASIC Magazine，5月号，89-90pp.

▶ HOT INFORMATION

ウィンドウシステム搭載！　X68000に新シリーズ登場。SUPER-HD，PRO II，EXPERT IIシリーズと，SX-WINDOWを紹介。――編集部，マイコンBASIC Magazine，5月号，91p.

▶移植版 PUUSUKE

ダイヤとハートを順番に取って家へ入る。パスワード入力でコンティニュー。――高橋潤，マイコンBASIC Magazine，5月号，164-166pp.

▶ SAVIOR X

自機を操って爆弾を順番どおり3つ回収し，時間内に脱出。爆弾回収ゲーム全6面。――古川泰之，マイコンBASIC Magazine，5月号，167-169pp.

▶ PE2

以前掲載されたパターンエディタのバージョンアップ版。クローン機能，複数パターンの合成などの機能が追加されている。サンプルゲームも収録。――宮本親一郎，ASCII，5月号，345-350pp.（ディスク収録）

▶ AV STRASSE

X68000の新機種について紹介する。SUPER-HDおよびEXPERT II/PRO IIのハードを始め，SX-WINDOW，ディスプレイ・光磁気ディスクユニットなどを取り上げる。そのほかサードパーティのハード，ソフトも登場。――編集部，ASCII，5月号，369-374pp.

▶ MiCRO MUSiQUES

Dynabook用MIDIインタフェイス，Mu-1，MUSIUM4といった音楽関連のソフト＆ハードの情報。――編集部，ASCII，375-376pp.

▶ X68000シリーズ「新ライン」ここが変わった

新たに発表されたX68000とウィンドウシステムSX-WINDOWの概要，メリット，その可能性についてレポートする。――高橋雄一，マイコン，5月号，156-164pp.

▶サンダーブレード with XE-1AP

アナログ入力対応のサンダーブレードを，電波新聞社のアナログジョイパッドXE-1APを交えて紹介。――あゆかわさつみ，マイコン，5月号，202-203pp.

▶なんでもQ&A

HumanのSWITCHで色変更を行ってもVSでは元に戻ってしまうのはなぜか，Sampling PRO-68Kの録音データをHumanのコマンドモードで再生するにはなどの質問に答える。――編集部，マイコン，5月号，362-363pp.

▶ X68000

シャープから発表されたX68000の新ラインナップを紹介する。変更点の概要や発売予定の周辺機器などを解説。――編集部，I/O，5月号，166p.

▶フォントユーティリティ

拡大文字を使ったり，テキストを任意のサイズで表示したい場合に便利な文字フォントサイズ変換ルーチン。――WIZARD N氏，I/O，5月号，177-182pp.

▶ DIRCOPY 改良版

ディレクトリ単位のコピーのためのユーティリティ。コピー元にファイル名を指定できるようになり，さらにコピー条件の種類が増えて再登場。――市原昌文，I/O，5月号，183-189pp.

▶ BATKEY

Human68kでのバッチファイルの操作のキャパシティを広げる。バッチ処理で条件分岐を行うプログラムだ。――牛島健雄，I/O，5月号，190-191pp.

ポケコン

PC-G801

▶誌上公開質問状

PC-G801のROM内ルーチンやアドレスマップについて解説している書籍を紹介。そのほかシャープポケコンについての質問に答えている。――I am，マイコンBASIC Magazine，5月号，88-89pp.

PC-E500

▶弓道

ポケコンで武道に挑戦。10回放って得点を競う。――Calling，マイコンBASIC Magazine，5月号，171p.

▶マネナンダ

沙羅曼蛇モドキのシューティングゲーム。――佐藤祐紀，マイコンBASIC Magazine，5月号，172p.

▶ポケットコンピュータ活用研究

PC-E500で表計算を行い，表データをPC-9801とやり取りすることを試みる。――塚田洋一，マイコン，5月号，321-326pp.

都市の書物

本書はほぼ全面的にコンピュータの管理下において執筆，制作，印字，編集，制作されている。いわゆる，最近注目を浴びているデスクトップパブリシング（DTP）によって出版されている書物だ。池澤夏樹氏の創作，批評，随筆，翻訳などにグラフィックデザイナーの戸田ツトム氏が構成をほどこしたひとつの作品である。もちろん読み物としても面白いが，本書を見ることによって最新のDTPを知ることに意義がある。

池澤夏樹著　戸田ツトム構成　太田出版
☎03(359)6262　A5判変形　119ページ　1,400円

ニューメディアの興亡

今日，コミュニケーションの道具として数多くのメディアが使われている。電話・テレビなどは当たり前として，それに加えてニューメディアなるさまざまな媒体が出現してきたのである。しかし，ニューメディアとはいえ，世の中に浸透するものもあればそうでないものもある。本書は，その興亡の裏側をビデオ，キャプテン，パソコン通信，衛星放送などを例にとって電波新聞に連載（102回）されたものをまとめたものである。

青木貞伸著　電波新聞社　☎03(445)6111　B6判　285ページ　1,500円

書籍の価格は消費税込みです

QUESTION and ANSWER

X68000PROを買って半年がたちました。ちょっと前に自分で配列に格納されているデータをソートするプログラムを作ってみました。数字の場合はそのまま大小を比較すればいいし，英字の場合も同様に文字列の大小を比較すればいいので，これで「完成」と思っていました。

そこで，これを利用して以前に作った電話帳を50音順に並べ換えれば便利だろうと思い，早速とりかかったのですが，思ったようにソートされないのです。英数字の場合はきちんと動いているのに，漢字の場合に結果がおかしいのです。フローチャートとプログラムリストを同封しますので，よろしくご回答のほどお願いします。

埼玉県　陶山　謙二

アルゴリズムは基本交換法（バブルソート）であり，対象件数が多い場合には時間的に不利ですが，アルゴリズムにもプログラムにも，おかしいところはどこにも見当たりません。そうでないとするとどうしてなのか？　さっさと原因をいってしまうと，たいしたことはない。JISの漢字コードの並びに問題があるのです。X68000の取扱説明書の後ろのほうに，JISの第1水準，第2水準の漢字コード表が掲載されていますからそれを参照しながら読んでください。

ここでちょっと簡単な試験をしてみたいと思います。あなたは自分の名字に使われている漢字をすぐに探し出すことができますか？　もしかしたら自分の漢字が見つからない！　という人もいるかもしれません。実際に私の友達にこの作業をやってもらったら，彼は「川」の字が見つけられなくて困っているではありませんか。確かにカ行で見ても「川」は見当たりません。音読みで探してみて，と私がアドバイスすると彼はやっと目的の字を探し当てることができました。「家じゃいつも『かわ』で川と（読みづらい文章だな）変換してくれるのに」と彼はいいましたが，それはワープロの使っている辞書に，川が「かわ」の読みで登録されているからであって，必ずしもその読みでJISコードに登録されているとはいえないのです。

彼の名前は川口といいますが（協力ありがとう），まさか自分の名前に使われている漢字が音読みでない限り，わざわざ音読みで探し始める人はいないでしょう。もちろんコンピュータで漢字を扱うことに慣れている人は，まずは音読みを考え，音読みがない漢字の場合は訓読みで探し始めますが。

このJIS漢字コードはコンピュータの中でそのまま漢字を表すコードとして使われることは少なく，一般的に広くシフトJISコードと呼ばれるものが使われています。たとえばJISコードの場合，ASCIIコードの中にJIS漢字コードが混じっていても，コンピュータはそれをわかってくれません。そのために，ここから先はJIS漢字コードだよ，とわからせるための漢字INコードと，JIS漢字コードはここで終わりだよ，とわからせるための漢字OUTコードが，それぞれ必要となってくるからです。

これらは本来の文字情報には必要のないものです。そこで，拡張ASCIIコード（本来のASCIIコードは7ビットですが，日本ではこれを8ビットに拡張して，カタカナやグラフィック文字などのキャラクタを追加している）の&H80～&H9Fと&HE0～&HFCを漢字コードの1バイト目に割り当て，ASCIIコードと漢字コードを区別するコード方式ができました。これがシフトJISコードと呼ばれるもので，Human68kやMS-DOS，CZ-8FB02などで採用されています。

さて，英数字の大小を比較する場合はASCIIコードで大小を判断するように，漢字の場合はシフトJISコードで判断されます。たとえば，「大塚」，「影山」の頭1字をシフトJISコードで表すと，

　大：$91E5_H$
　影：8965_H

となります。これを見ても明らかなようにソートした結果は，

　影山
　大塚

となってしまい，我々の期待を見事に裏切ってくれます。

JISコード表を見ていて気づいたと思いますが，基本的に第1水準漢字は音読みで50音順に並べられ，第2水準漢字は部首別に分類されています。そのため名字の読みが音読み（がない場合は訓読み）でない限り，（人間から見て）正しいソート結果は期待できません。

では，いつものように対応策を考えてみることにしましょう。このプログラムの場合，うまくいかない原因はソートする項目（特にこれをキーという）が漢字だということです。そこで別にキーを用意することを考えてみたらどうでしょうか。

英数字ならちゃんとソートできるのですから，名前をローマ字で表した配列を新たに作ればいいような気もします。しかし，よく考えてみるとそれも駄目なことがわかります。では，いったいどうすればいいでしょうか。

もう一度漢字コード表を眺めると，非漢字の全角文字コード表にある平仮名（片仮名でも）がきちんと50音順に並んでいます。どうやらこれを使えばうまくいきそうです。

```
name (1)＝“影山裕昭”
k (1)＝“かげやまひろあき”
tel (1)＝“01-234-5678”
```

のようにして，配列kをキーとすれば万事解決します。お試しください。

X1でマルチウィンドウみたいなことをやってみたいのですが，できるものなのでしょうか。採用されるまで何回でもハガキを出します。うひょ。

東京都　安藤　正雄

なんと大胆かつ奇妙な質問（文面）なんでしょうか。変なヤツ，と思ったんだけど，なかなかおもしろい質問なので取り上げてみました。で，マルチウィンドウみたいなもの，は比較的簡単に作ることができそうです。まず，BASICを立ち上げて，画面表示を80桁にしたら，

```
console 3,15,10,50
```

として，

なにかプログラムを読んでリストを出し

てみてください。表示される範囲が小さくなっているでしょう。これはconsole命令で，表示エリアを（10,3）－（60,18）に設定しているからなのです。そのままで今度は，

console 8,15,30,50

としてみましょう。さっきと同じようにリストを出すと，表示エリアが右下に移動しましたね。console文の使い方はリファレンスマニュアルを見てもらうとして，この表示エリアの外側に枠を表示して，ウィンドウのあいだを自由に行き来できるようにして，さらにウィンドウの大きさも自由に変えることができれば，ある程度マルチウィンドウ「みたいなもの」にはなります。

しかし，これではマルチウィンドウに欠かせない大切なものが抜けているのですが，お気づきですか。いまの例のように，上に表示されているウィンドウの下に隠れているウィンドウに表示されている情報を，どこかに保存しておかないと，下のウィンドウに戻ったときに一部の画面情報が欠けてしまうことになります。ですから新しいウィンドウを開くときには，現在アクティブなウィンドウの中身をどこかに保存しておく必要があるのです。

さらに，うっかりすると保存し忘れてしまいそうなのが，アトリビュートの情報です。単色でCGEN命令を使わないならアトリビュートが書き変えられることはありませんが，それ以外の場合はテキストRAMと同じくバッファにデータを保存しなければいけません。

また，複数のテキストを同時に編集するようなこともできません。

これぐらいだったらプログラムにすることはそう難しくないでしょう。グラフィックを使用しないことを前提に置けば，G-RAMをウィンドウのバッファに使えばいいし，ウィンドウを開くキーは以前質問箱でも取り上げたように，未使用のコントロールキーに割り当てればよさそうなものです。面白そうなのでサンプルプログラムを作ろうと思ったのですが，サイズが大きくなりそうなのでやめときました。機会があればまた別の誌面で紹介しましょう。

構造化プログラミングという言葉をよく耳にするのですが，プログラムを構造化すると，どんな長所があるのでしょうか。

神奈川県　鈴木　昭雄

いままで適当にプログラムを作っていましたが，こんど真面目に大作プログラムといえるようなものを作ってみようと思っています。そこでプログラムを構造化しようと思っているのですが，その上で気をつける点があれば教えてください。　新潟県　石田　智樹

似たような質問なので2人まとめて回答しましょう。構造化プログラミングによる利点は，誰が見てもわかりやすいプログラムだということでしょう。

構造化プログラミングを提唱したダイクストラさんは，連接，判断，前判定反復の3つの基本制御構造を組み合わせれば，すべてのプログラムが書けるとおっしゃいました（実際には後判定反復などもあるが）。これらの基本制御構造は原則的にひとつの入り口とひとつの出口を持っていて，出口まで行かないで途中で抜けたり，入り口以外のところから入ってくることを禁止しています。ですから，ループの途中で抜けたりすることは当然禁止。また，むやみやたらにGOTO文であちこちに制御を移すことも，プログラムを理解しづらくするという理由で，極力使わないようにしたほうがいいとされています。

それにプログラムをモジュール化することも大切です。モジュールとはある大きなプログラムの中で，独立した機能を持っているものを，そのプログラムから独立させてモジュールごとに開発していこうというものです。モジュールに渡される引数はほかのモジュールに影響されないことが望ましく，さもないとモジュールの独立性が失われる結果となってしまいます。しかし，いくらモジュール化するといっても，

初期設定
↓
メイン
↓
終了処理

と素直に連接しているプログラムを，

gosub 初期設定
gosub メイン
gosub 終了処理

としているのもありますが，わざわざこのようにする必要はありません。

モジュールごとに分割されているプログラムはあとでバグが見つかっても，されていないプログラムに比べて，デバッグに費やす時間が短くてすむという長所があります。

マルチステートメントはプログラムの区切りをわからなくし，プログラムの流れをわかりづらくするものですから，なるべくやってはいけません。Cでは「1行には1文」というのが定着しています。昔，コンピュータの記憶容量が少ないころは，メモリを効率的に使うためにマルチステートメントが多用されていましたが，そのためにマルチステートメントを多用するんだったらナンセンスです。また，実行速度を少しでも速くするためであったら，コンパイラを使うか，初めからマシン語やCを使えばいいのです。　（影山　裕昭）

質問にお答えします

日ごろ疑問に思っていること，どんなことでも結構です。どんどんお便りください。難問，奇問，編集室が総力を上げてお答えいたします。ただし，お寄せいただいているものの中には，マニュアルを読めばすぐに回答が得られるようなものも多々あります。最低限，マニュアルは熟読しておきましょう。質問はなるべく具体的に機種名，システム構成，必要なら図も入れてこと細かに書いてください。また，返信用切手同封の質問をよく受けますが，原則として，質問には本誌上でお答えすることになっていますのでご了承ください。なお，質問の内容について，直接問い合わせることもありますので，電話番号も明記してくださいね。

宛先：〒102 東京都千代田区
九段南2-3-26井関ビル
㈱日本ソフトバンク出版部
「Oh！X質問箱」係

FROM READERS TO THE EDITOR

4月から新生活をスタートされた人たちも多いことだろうと思います。しばらくたちましたが，学校や仕事には慣れられたでしょうか。まあ，悩んだときなどには，Oh!Xでも読んで気分を一新してがんばりましょう。

◆3月18日，日曜日。毎月のように本屋に行って，いつもOh!Xの置いてある場所へ行った。……しかし，Oh!Xが見あたらない。「そうか，今日は日曜日だ」。そして次の日。しかし，ない！　その次の日も……。不思議に思って店の人に聞くと「入ってますよ」と，とんでもない雑誌を持ってきた，と思いきやア然「えっこれはLOGINじゃなかったのか……」。　横川　要（18）長野県

「3月18日の衝撃」はどうでしたでしょうか。新しい表紙についての感想は……。

◆表紙がイカス〜！　CGがイカス〜！
河野　太郎（17）東京都

◆新しい表紙はなかなかいいですね。なんか高級感があふれていて1,500円ぐらいの本を買ったようで得した気分です。
竹永　昌伸（16）兵庫県

◆新しい表紙はかっこいいですね。あれはひょっとしてX68000で描いたんでしょうか？
田中　義人（20）福井県

あれはPC-9801にフレームバッファを積んだものです。しかしX68000で表紙を描いている雑誌もあるようですよ。

◆表紙が新しくなったけど「おえかきソフト」で描いたグラフィックは避けてもらいたい。
新畑　貴史（19）広島県

どこに出しても通用する本格派のCGを載せますので，ご期待あれ。

◆4月号の表紙の右と左後ろに見えている建造物は「原子力発電所」ですか？　また手前の妙な装置（なんとなく武器のように見える）の名前「ARES」とは「戦いの神」じゃないですか！　う〜ん，このCGを描いた須藤さんって怖い。
渡辺　久孝（23）岡山県

須藤さんは原発に似ていると言われ「あっ，言われてみればそうですね」などとおっしゃっていますけど。ちなみに原発のように見えるものはビルで，前面にあるのは捨てられた機械（でも実は生きている）だそうです。

◆新しくなった表紙を見てうちのおっかさんは「家の光」（農協が出版しています）という本と間違えた。　安尾　文教（22）愛知県

表紙が変わったのでこれで「ムー」と間違えられることはなくなったと思っていたのに……「家の光」とは。ちなみに報告があっただけでも「Oh!PC」，「マイコン」，「Oh!FM」，「LOGIN」，「トランジスタ技術」，「某誌」……などと間違えられたらしいです。なお感想は，頭がよさそうに見える，very good，カッコイイ，汚れが目だつ，こすると色が落ちる，ツルツルしてる，コンピュータ雑誌らしくていい，白くて寂しい，前のほうがよかった，どうして変えたんだ，元に戻せ！　などさまざまでした。ただ大半の読者の反応は「イイ」ということなのでよかった，よかった。

◆特集はゲームを買うときの参考になり，とてもよかった。　松野　竜明（17）愛知県

◆ゲームは新しいメディアとして確立するか，これまでの娯楽道具から進化するかなどと考えてしまった。　森田　謙一郎（17）鹿児島県

両方生き残りまっせ。（吉田氏談）

◆特集ではゲームの「攻略」ではなく「内容」を見せてくれたのがよかった。
梅津　長裕（17）東京都

攻略を書いてという読者も多いんですよ。

◆特集を読んで久しぶりにゲームがやりたくなってしまいました。　山口　幸一（23）東京都

◆乏しい小遣いのなかからポピュラスを購入してしまった。ダンジョンマスターもまだ途中だというのに……。しかしこのゲームも面白い。本当に個人の性格がはっきり出てしまう。私は思わず自分のあまりのアコギな……あとは書けない。しかし敵もさるものでかなり報復してきます。　佐々木　元（41）東京都

U氏もポピュラスは「危険なゲーム」だと言っていますが……。

◆「ダンジョンマスター」が欲しい。欲しいけど金がない。金がないけど欲しい。ということで，あと残るは銀行強盗しかない。では，さよなら。　横川　聖一（17）兵庫県

今度横川君に会うのは新聞の上だったりして……。

◆「ゲームミュージックを斬る」ですかぁ。僕もあの初代V.G.M.に感動したものです。特にギャラガなんか……。あといちばん好きなのは12インチシングルで出たスーパーゼビウスとギャプラスにドルアーガの塔……。う〜ん，最高っ！
飯田　範明（19）神奈川県

◆「スーパーハングオン」をやる。トラックに乗ったおじさんが「カメー」と叫びながら抜いていく。頭の上を天使が飛んでいる。木にぶつかってこけると羊さんが集まってくる。コースの横にステゴザウルスとイルカとペンちゃんがいる……。ああ，どんどん怖い考えが浮かんでくる。　藤原　利治（23）東京都

ゲームプレイは個人の世界。どんどんのめり込みましょう。ゲームだったら飲酒運転だって許される……かな？

◆ヴァリスの優子の目は超でかい！
長谷川　敬之（17）福岡県

たしかに。

◆「エアーバスター」というアーケードのゲームには「X68000（マンハッタンシェイプ）ビル」や「TOWNSビル」が出てきます。作ったのはどんな方でしょうか。　川崎　修治（23）兵庫県

◆「第4のユニット4 Zerø」の記事で「絶対買う

◀川上　員護　埼玉県
引っ越しで出てきた昔のハガキだそうですが，たしかに40円ハガキというところとハガキの黄色さがその古さを物語っています。

◀福原　徹　埼玉県
うん，変わりましたよ。まあ，前のほうがいいとか良くなったとか，いろいろ意見はありますが，それぞれの好みの問題ですからねぇ。

ぞ」。古村さんに感謝します。

増田　勝（20）大阪府

ブロンウィンの記事からは（で）氏の思い入れがヒシヒシと伝わってきますからね。

◆「銀河英雄伝説」の説明をしていた「人気ゲームのシステムを探る」の解説者はすこしきびしすぎる。　竹内　知二（18）大阪府

でも、銀英伝の評価はまずまずですね。

◆「GAME OF THE YEAR」の「海外移植ゲーム賞」を見て思った。来年この部門に書くとしたら……。ダンジョンマスター？　ポピュラス？ゲームを始めるとどちらも「こいつがいちばん面白い」と思ってしまう。これでシムシティが出たら……。　杉本　尚敏（20）大分県

最近X68000で動く面白いゲームは海外のものが多いですね。日本のソフトハウスにも頑張ってもらいたいものですが。

◆「GAME OF THE YEAR」の主演キャラクター賞「テトリスの直線ブロック」はウケた。16票も入っているなんてねぇ。

津金　尚志（16）埼玉県

意表を突いたようで実はみんなが思っている結果でしたね。

◆今月のゲームシステム文学誌は今までのどのゲーム特集よりも「力」が入っていたと思いました。なかでも荻窪さんの「ゲームソフト進化論」はとても面白かった。

功刀　和久（20）埼玉県

サイバーでハイパーでウエットなゲームが誕生するだろうか。

◆X68000のゲームソフトはかなり質のいいものが多いですが，ユーティリティや３種の神器に関してはイマイチな感じがします。サードパーティからもっと発売されないかなと思うのは僕だけではないはず。

矢島　裕仁（22）神奈川県

カラーテレビはナショナル，車はトヨタ，クーラーは日立ですからね。早くシャープからも出してほしいですね。（ボツ）

◆聴くに耐えなくなった歌謡曲に見切りをつけて，私はゲームミュージックを聴くようになった。しかしゲームミュージックが一般に認められるようになるにはまだまだだと思う。ハズレソフトの割合が大きすぎる。

桑原　敦（20）東京都

昔，映画音楽からヒット曲が生まれ続けた時代がありました。将来のゲームミュージックもそうなるかもしれませんよ。

◆イースIIIを買いました。面白くて１日でガルバランまで行ってしまった。しかしそれからガルバランのHPのグラフが１cmぐらいしか減らない。どうにかして！　浜口　航介（13）岐阜県

編集部のみんななんか0.2ミクロンぐらいしか減りませんよ（ウソ）。

◆「便利なsp_chk( )」は役に立ちそうです。でも考えたんだけどX-BASICって不便な点も多いですねぇ。　萩原　保憲（23）神奈川県

確かに不便なところもありますが，拡張性に優れたシステムだからこそ「sp_chk( )」

◀大野　真実　静岡県

あぶねぇー、こんなのが出たらユダヤ人が団体で怒ってきますよ。「天下どどいつ」というから、都々逸でも唸るのかと思いましたが。

◀小林　貴洋　千葉県

お父さんの顔が地獄の閻魔様みたいでこわいですね。まあ、大学に受かってから何台でも好きなだけX68000を買えばいいじゃないですか。

のような対応ができるんですよ。

◆「The Cave of Dalk」は予想をはるかに越えた素晴らしいゲームのように思えました（タイトルもいい）。さっそくMZ-2200に入力してプレイしてみようかと思っているところです。

庄島　賢一（21）東京都

◆ファジィコンピュータシミュレータでなにをすればいいのですか。

山中　雅彦（29）新潟県

なんでもできますよ。人工知能とか，郵便番号認識とか，洗濯とか……。

◆こんど絶対フロッピー付けてくれ!!

岩浪　吉高（19）神奈川県

◆Oh!Xにもフロッピーディスクの付録を付けてほしいよぉ〜。　藤巻　康昌（16）静岡県

という声は前々からありました。全国100万のOh!X読者の方，お待たせしました。付きました，付きましたよ。ディスクが付きました。内容はほかの雑誌の500倍充実を目指した実用的なツール群。なにが入っているかは開けてビックリ玉手箱（なんかジジくさいなぁ）。これは買うとすれば数万円は下らないシロモノですぜ，ダンナ。

◆ディスク版の付録があったらとてもうれしいのに……。だからといって定価が上がると悲しい。　株本　義久（15）大阪府

特別定価の780円は「見逃してくれよ」。

◆３月16日　なにげなく新聞を読んでいたら小さな記事が目に入った。見出しには「シャープが和製Macintoshを発売」といったようなことが書いてあった。その記事だけではどういうものかわからないのでOh!Xを楽しみにしている。

若林　勝（21）大阪府

◆３月25日，私は秋葉原に行き，某ショップで妙なものを見つけた。こっ，これは!!　X68000ニューラインアップの広告ではないか（う〜ん，SUPERってダサイ）。私はさらにヘンなヤツを見つけた。SX-WINDOWだとぉ〜。ひとつのウィンドウにはVS.Xのようなファイルアイコンが，別のウィンドウにはカレンダーが，さらに別のウィンドウにはおねえさんと犬が走るアニメーションやら宇宙のグラフィックやら，なんとピンボールのゲームまで。「こいつぁ春から縁起がいいや」と思った幸せな日だった。

栗坂　明（21）埼玉県

◆秋葉原でSX-WINDOWとHyperwordのデモをやっていたのでいじってみた。Hyperwordはイマイチという印象，SX-WINDOWは見た目はスゴイが，はたしてウィンドウインタフェイスとしてどれほどの力を持っているか興味深い。

宮脇　信行（22）東京都

◆SX-WINDOWってぱっと見がなんとなくNeXTしていますね。　越智　博一（26）山口県

◆SHIFT BREAKのＴ氏の予想どおりシャープはフェイント攻撃に出たようですね。

白井　保弘（21）三重県

◆いやぁ〜。まいりましたね，EXPERT IIとSUPERには。EXPERT-HDを注文していたのですが急いでキャンセルしました。それにしてもSX-WINDOWはどんなものなんでしょう。早く使ってみたいな。　仁藤　浩明（16）山形県

SX-WINDOWの編集部での評判は上々ですよ。しかし，SUPER-HDは某ビデオテープみたいな名前だという意見もありますけど。

◆大ショック！　X68000SUPER-HD　次に出るならSCSIインタフェイスは装備されるのではないかと思っていたが，80Mバイトのハードディスクが内蔵されるとは予想できなかった。しかも強力なウィンドウソフト付き。これで３万円

▲清水　健年　東京都

ありがとうございます。100号を前にして廃刊ということはたぶん（絶対？）ないと思いますが，気を許さずにがんばります。

の価格差しかないなんて，技術の進歩は罪なものよのう。　横田　紀明（23）山口県

◆12月にX68000を買ってからはや数カ月。今日も私の部屋からはディスプレイをみがく音が聞こえる……フフフ……。
堂領　輝昌（16）宮崎県
明るい光景だといったほうがいいのか暗いといったほうがいいのか？

◆やった～，第1志望に合格しましたぁ。というわけで昼はダンジョンマスター，夜はバイトという生活を送っています。
山田　直（18）愛知県

◆あの女の子は受かった。あのスケコマシの友達もその女の子と同じ学部，学科に受かった。とても悲しい。あの女の子はすべった。同じ予備校のクラスになれそうだ。その女の子の彼といっしょに……。ひじょ～に，悲しい。
佐藤　能久（18）大阪府
大学は遊びの場所じゃない！　なんてOh!X編集部で言っても説得力ないかなぁ。

◆1から9までの数字を1回ずつ使って，3ケタの数字を3つ作ります。それを足して9で割ると必ず割りきれます。これを数学的に証明せよ!!　大久保　益幸（18）滋賀県
だから，987+654+321=？？？でしょう。それを9で割ると……，うっ，あっ頭が割れそうだぁ。

◆最近空を飛びたくなった。「アフターバーナー」は平和じゃないし，ゆったりとしたものをやりたいな。　宗宮　良直（15）愛知県
私も逃避したい。

◆サークル仲間とスキーに行ってきました。3泊して3夜とも宴会。寝不足気味と二日酔い気味で滑り，また飲む。悲惨な日々であった。
加藤　信之（20）東京都
えっ，悲惨な日々？　酒とバラの日々じゃないですか。うらやましい……。

◆コーヒーはUCC，英会話はECC，CコンパイラはGCC。　井戸　直樹（19）岐阜県
まだあるぞ，通信制御機構はICC，広告の賞ACC，某ソフトハウスはNCC……。

◆僕がさっきヒマつぶしにトランプの札をよくきって黒を出すのを念じていると恐ろしいことが起こりました。なんと6回連続で黒が出たのです。それは1/64の確率なのです。やはり人間にはPOWERがあるのだろうか……。
上リロ　晃成（16）兵庫県
そいつぁ，スゲエや。人知の及ばない宇宙の深淵をかいま見たぜ。

◆私つい先頃ロンドンに行ってきました。ロンドンは伝統と自然と活気にあふれたすてきな街です。皆さんヨーロッパに行くならロンドンです。パリやローマに目がくらんではいけません。
中嶋　祥史（22）神奈川県
私なんて，「愉快なロンドン」や「羅馬」に行ったことあるし，しょっちゅう「パリジャン」食べてるぞ。えっ，自慢にならないって？

◆4月号のゲーム特集で，カラーページがどうも読みにくいと思ったら半透明機能を使っているじゃないか！　X68000じゃあるまいし……。
木村千映（21）東京都

◆ゲテモノドリンク専用のコーナーを作ってください。タイトルは「ウルトラ怪汁大図鑑」がいいな。　河辺　義信（18）愛知県

◆画像取り込み・処理の記事なんてどうでしょう。ソフト化，モザイク化，モノトーン，拡大，ミラー，LUTなど。　伊藤　裕幸（28）東京都

◆3/24（土）に任天堂と電通によるゲームセミナーに参加してきた。テレビの生中継があったり，「ゲーム業界にこの人あり」と言われる有名な人たちに会えたりしてとても心がワクワクしました。このうち何人かが講演してくれたのですが，いちばん興味を持って聞くことができたのはやはり大ボラを吹いて帰っていった糸井重里さんでした。彼はイイ人だ。
太田　敬三（20）東京都
で，業界からの引き抜きはなかったのですか？

◆某LOGIN誌みたいに「日本ソフトバンクOh!X編集部」を特集してもらえないだろうか？　見てみたいのですが……。それとも，見せられないほどのところなのだろうか？
清水　義弘（21）沖縄県
いやぁ，見せるホドのところじゃないだけですよ。

◆私が就職する会社ではTITANと呼ばれるコンピュータが動いている。なんと，それに積まれているハードディスクは1Gバイトでメモリは320Mバイトなんだそうだ。しかもまだメモリが足らないのだそうだ……（絶句してしまった）。
古田　雄一（21）埼玉県
でも私の知っている人はTITANを個人で買ってしまったんですよねぇ。これホントの話。

◆祝！　吉田幸一氏復活！　なぁなぁまぁまぁのパソコン業界を大人の視点で斬ってほしいと思っています。　溝口　伸一（21）東京都

◆「ぜんまいちゃん」の復帰を心からお喜び申し上げます。　池崎　善久（25）東京都
吉田幸一氏いわく「吉田戦車の女の子爆弾シリーズはぜんまいちゃんのパクリだ」。

◆吉田幸一氏の復活とても喜ばしく思います。ところで「スタッフ」として名前の出ていない祝氏や吉田幸一氏はどういう立場にいるんでしょうか？　倉田　泰幸（20）茨城県
さぁ，彼らにもいろいろと都合があるみたいですし。しかし吉田幸一氏の「復活」にはお祝いのハガキが殺到しました。ぜんまいちゃん，呪一平，K君，などが再度登場すればいいですね。

◆高校の後輩に電脳倶楽部なるものを見せてもらいました。そうか，実在していたんですね，「満開の電子ちゃん」は。あまりにも嘘っぽかったのでOh!X特有のアレかと思っていたのですが……。　中村　健（20）埼玉県
Oh!Xは嘘は申しません。ぜひ清き1冊を。

◆Oh!Xがシャープ公認の本とは知りませんでした。すごいですね。これからも頑張ってください。　露崎　達也（17）千葉県
べつに公認というわけじゃないですよ。たぶん推薦くらいならしてくれるかもしれませんが……。

◆ホント「ため」になる本です。月1回ってのが許せない。日刊にしても俺は全部買うぜ！
前田　光輝（18）千葉県
日刊にしたら，月16,800円ですよ。ホントに買ってくれますか？

◆ちょっと値段が高いのでは？　せめて400円台ならば毎月買えるのに。
柏木　仁志（18）栃木県
Oh!Xのコストパフォーマンスの高さは業界一ですよ。ぜひ毎月買ってください。

◆1989年12月にX68000を買ってから，Oh!Xも必ず買うようになりました。日本ソフトバンクの営業部に聞いてみてください。ここ5カ月売り上げが1上がっているはずです。
安藤　道子（？）宮崎県
Oh!X編集部特別調査室が調査したところ宮崎県方面でこの5カ月，1冊売り上げが増えていることを確認しました。

◆つい先日，我が家にアスリート以来の「毒物飲料」がやって来た。それはOh!Xでも有名な青臭いアルファルファ飲料「アルファエー」であった。しかも，1ℓパック×2という泣く子も黙

◀川島　裕一　京都府
X68000版Wizardryって出るんですかねぇ，うーんわからない。それはともかく，Vampire Lordって女の人だったんですか，知らなかったなぁ。

◀丸藤　俊之　神奈川県
またまた丸藤さんの作品です。相変わらずうまいですねぇ，さすがイラスト大賞をとっただけのことはあります。

る量であった。そこで一家揃って試飲したところ何と父や弟は「ドリンク剤と思えば飲めるじゃない」などとほざいたのです。「TETSUやキビジュースを飲ませたろか？」と思ったのは言うまでもない。3日でなくなったという恐ろしい事実も付け加えておこう。P.S.肩書きの「妙味（あとあじ）さわやか」には笑った。

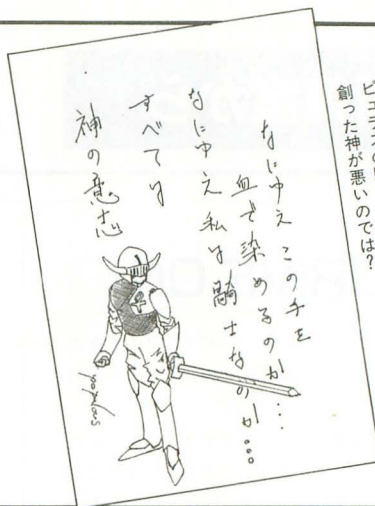
向坂　和博（20）埼玉県

読者が持ってきてくれた「アルファエー」(40缶！)も結局すぐになくなってしまいました。

◆最近忙しくてSTUDIO Xしか読めん。

原　政徳（17）愛知県

て，今月のラストは原君ですよ。ほかの記事も読んでね。

◀竹永　昌伸　兵庫県
スタージャッカーですか，私は残念ながら知りません。でも，一度見てみたい気がします。どんなゲームなんでしょうか。

◀後藤　正和　大阪府
台詞がなんか時代劇みたいですけど，なかなかピュラスの騎士の立場を的確に表していますね。創った神が悪いのでは？

ぼくらの掲示板

●掲載ご希望の方は，官製ハガキに項目(売る・買う・氏名・年齢・連絡方法……)を明記してお申し込みください。
●ソフトの売買，交換については，いっさい掲載できません。
●取り引きについては当編集部では責任を負いかねます。
●応募者多数の場合，掲載できない場合もあります。

仲　間

★X68000Onlyのサークル「号泣」。活動内容はゲームを中心に，ゲーマーの集える活動を目指して月1回「ゲーマーズ号泣」と題したディスクマガジンを発行。もしかするとゲーマーかもしれない君たち，X68000ユーザー諸君，きたれ我らがサークルに！　連絡は62円切手同封のうえ封書で。まってるぜ！　〒594 大阪府和泉市鶴山台団地1-713　西村美智子（19）

★パソコンユーザー(主にシャープ機種)，テーブルトークRPGファンを対象としたサークル「インフォメイション・ソサイエテイ」では，会員を募集しております。活動内容は隔月1回の会報「まじふぉめ」発行，ソフトの共同開発，ダンプリストの分担入力などです。興味のある方は，住所，氏名，年齢，パソコンを持っていれば明記のうえ，簡単なお手紙とともに62円切手を同封してご連絡ください。〒006 北海道札幌市手稲区富丘1条7丁目3-15　飯田伸一（16）

★M-SPEEDでは，X68000ユーザーの会員を募集中です。活動は月1回のDMで，内容は圧縮をかけた最新PDS，CG，MUSIC，GAME情報他，1Mバイトめいっぱい詰まったDMです（他PDSは大量所有）。連絡は300円切手(DM付き郵送料込み)または62円切手同封でお願いします。〒610-01 京都府城陽市寺田樋尻69-60　鶴野周吾（20）

★X68000ユーザーを対象としたサークルを発足させるにあたって会員を募集します。初心者大歓迎。興味のある方は62円切手同封のうえご連絡を。〒606 京都府京都市左京区吉田近衛町26　蟻田伸（18）

★X68000ユーザーを対象とするサークルを結集したいと思います。とりあえずは情報およびPDSの交換を主としてやっていこうと思います。初心者から上級者まで誰でも大歓迎。詳しいことは120円切手＋ディスク＋封筒を入れて封書で。〒503 岐阜県大垣市西大外羽3-89-2　後藤義宏（16）

★X68000ユーザーを対象としたサークルを発足させるにあたって会員を募集します。活動内容はソフトの必勝法やゲーム大会などいろいろと考えています。また，会誌も発行しようと思っています。興味のある方はまず62円切手を貼った封筒を同封のうえご連絡ください。詳細をお送りします。〒729-01 広島県福山市松永郵便局私書箱39号　後藤康豪（18）

★このたび「SUPER REDBUS」では一緒に活動してくれるスタッフを募集します。現在オリジナルゲームソフトを製作中で，主にX68000・PC-9801のユーザーの方でプログラミングに自信のある方，やる気のある方で都内近郊にお住まいの方に限らせていただきます。詳しいことは，62円切手2枚を同封のうえご連絡ください。〒170 東京都豊島区巣鴨4-22-3　高頭雅之（24）

★データサービスでは「X68000ユーザー友の会」を運営し，現在新規会員の入会申し込みを受け付けています。活動内容は会員証の発行，ユーザーズプレスの発行，ディスクサービス，etc……。入会希望の方は180円分の切手を同封のうえご連絡ください。〒616 京都府京都市右京区花園土堂町7　丹治文一（28）

★私達「スタジオ・ルクシオン」ではX68000用のゲームを開発するにあたりプログラマを募集します。愛知県内に住む「我こそは！」と思う方，ご連絡ください。その他のスタッフも募集いたします。まずはお手紙で。〒466 愛知県名古屋市昭和区川名本町6-33　酒井宗平（19）

★CG作成協力者募集！「シアター2＋1」では来たる8月下旬に下北沢駅前劇場において演劇とCGをドッキングさせた「The Silence of The War」を公演します。つきましてはCG作成にご協力いただける方を捜しています。詳細は劇団までご連絡ください。〒190 東京都立川市柴崎町2-12-10　シアター2＋1　いさらい香奈子（22）

売ります

★X1用FM音源ボード「CZ-8BS1」＋αを送料込み7千円で。箱なし・マニュアルあり・付属品あり。連絡は往復ハガキで。〒116 東京都荒川区町屋5-17-35　武田伸一（20）

買います

★X68000用増設RAM「CZ-6BE1」を2万円以内で。完動，付属品付きなら多少のキズ可。送料こちらもち。連絡は往復ハガキで。〒963-01 福島県郡山市安積町笹川字中の渡戸34-28　佐々木健（17）

★X1用マウスを3千円，X1turboII用第2水準漢字ROM（CZ-8BK4）を3千円，「試験に出るX1」「X1-Technow」をそれぞれ1500円で。また1200bpsのモデムを5千～7千円，2400bpsなら＋1万円で。連絡は付属品の有無，希望価格を明記のうえハガキで（モデムはメーカー，製品番号も明記してください)。〒178 東京都練馬区大泉町2-32-20　佐藤貴之（19）

★MZ-1500を送料込みで1万円，完動品のこと。往復ハガキで。〒719-11 岡山県総社市井尻野1646-10　天野浩（33）

バックナンバー

★Oh!MZ1987年1月，2月号を送料込み各千円で，連絡は往復ハガキで。切り抜き不可。〒247 神奈川県鎌倉市大船2062-5　高橋智史（17）

★Oh!X1989年1月号，2月号，3月号を送料込み2千円で。「X68000活用研究II」を2千円程度で。連絡はハガキで。〒671-12 兵庫県姫路市勝原区山戸241-10　山根邦博（16）

編集室から from E・D・I・T・O・R

DRIVE ON

このコーナーでは，本誌年間モニタの方々のご意見を紹介しています。今回は，4月号の記事に関するレポートです。

●ゲーム特集についてですが，特にゲームシステムとしてひとまとめにするよりは独立させてもよいと思うし，統一的なテーマもあまり見られないために位置づけや最終的な詰めの甘さが見受けられた。ゲームシステムに対しての考え方がバラバラというのは読者にとってあまり面白いものではない。むしろ1つひとつは単なる個人的主張のオンパレードとして，「システム」としてよりはその「メカニズム」や「ポリシー」にこだわってほしかった。特集とするからには常に明確な「テーマ性」を提示しておく必要があると思う。

中野賢一（18）X1/G/turboII, MZ-2000, FM-8, PC-9801ES5/N, FP-200, PC-1251, B16X 山口県

●ファジィコンピュータについては常々関心を持っていたため，非常に面白く読ませてもらった。現実にこれをどう使うかということはおいといて，こういった科学技術で知的好奇心を満たして遊べるような記事を増やしてみるのも面白いのではないだろうかと思う。

西田宗千佳（18）X68000, X1Fmodel20 千葉県

●新しい表紙に変わってとてもスッキリしましたね。Oh! PCに似てる気もしますが，よいと思います。ところで表題とバックの色はずっと白と黒なのですか。たまには変えてみてください。また，表紙絵がCGになったのですから読者の作品で飾ってもよいのではないでしょうか？

藤原博人（26）X1turboZ 鳥取県

●(で)のショートプロぱーてぃでは外部関数sp_chk( )というのが紹介されていましたが，今後もどんどんこういった「部品」を集めていって最後にシューティングゲームみたいなものでもできれば面白いと思います。

田中実（19）X68000ACE, X1turboII 大阪府

●X1およびMZユーザーにとっては読む記事が格段に減ったと思います。X1やMZ専門の記事ではなくてもよいから，アルゴリズムやオブジェクト指向だとかそのようなものの考え方を記事にしてもらいたいです。アルゴリズム講座なんか本当にやってほしいです。

末吉克行（21）X1G, MZ-731, FM-7 兵庫県

●「知能機械概論」，本が好きな私にとって今月号で最も面白く読ませていただきました。私はペーパーレス時代は絶対来ないと思います(単に来てほしくないだけかな。私は紙が好きなの)。なぜかというと，これは時代の流れでもテクノロジーの進化（発達かな）にも関係ない。人間そのものの本質によるものだと思えるのです。情報がソフト的に扱えるようになればなるほどそれらをはっきりした「形」として求めたがる。それが人間の物質交換の欲求だと思うのです。まあ将来どうなるかは我々の子孫しか知らないわけですが。

大津和之（20）X1turboZ 福岡県

●まあ予想されたとおり全般にX68000のソフトが占め「アフターバーナー」が多くの賞を取りました。順当勝ちでしょう（私にしてみれば面白みに欠けるような気がしました)。その中にあって「ねじ式」の存在は大いに注目しました。このようなオリジナリティのあるソフトがもっと出ればいいと思いました。私もこのソフトを知って書店でつげ義治氏の本を探したものです。それにしてもX68000以外のソフトの少なさがたいへん気になります。こちらでもX1のソフトを取り扱っている店は消滅してしまったし，もうだめなのでしょうか。GAME OF THE YEARはそういう意味ではパソコンの発展史のような気がします。このコーナーを見ればどのようなパソコンが隆盛しているかがわかるのです。このコーナーからX1のソフトが消えず，今後ともX1が発展することを期待しています。

森川一（24）X68000ACE-HD, X1turboII 北海道

＊

さて，先々月から募集してきました第6期愛読者年間モニタですが，なんと4月25日現在，応募者が7名しかおらず，今月は採用者の選出と発表を見送らざるをえない状況です。

X68000のユーザー数が伸び，おかげさまで，本誌の部数も順調に推移していますが，読者の皆さんの参加意識が以前に比べて薄れ，受け身になっているのではという指摘もあります。確かにユーザーの皆さんを取り巻く環境は豊かなものになりました。メーカーのサポートもよく，市販ソフトも数多く発売されています。しかし，やはりパソコンの世界はユーザーの手によって作られていくのが望ましい姿ではないでしょうか。

Oh! Xの雑誌づくりに「意見しちゃおう」という方，DRIVE ONのコーナーを通して本誌読者に訴えかけたいと思う方の参加をお待ちしています。（編集部）

ごめんなさいのコーナー

4月号　STUDIO X

P.171 掲載されたメッセージと氏名が食い違っていました。中段下「◆Oh! Xに取り残された～」のお名前が「鈴木茂さん」となっていましたが，正しくは「浜地啓さん」の誤りでした。謹んでお詫びいたします。

5月号　グラナダ

P.36 ウルフ・チームの問い合わせ先の電話番号が間違っていいました。正しくは，

03(5273)4795

でした。お詫びして訂正いたします。

5月号　豪華版SCRAMBLE

P.81 リスト2の300行に不適切な部分がありました。通常のドライブ設定で利用される場合は，

```
  300 img_save("f:map"～)
→ 300 img_save("map"～)
```

のように変更してください。

5月号(で)のショートプロぱーてぃ

P.148「夜中にひとりで見てはいけないデモ」をコンパイルする際のオプションは本文で表記したとおり，必ず大文字で指定してください。

CC/O/Y/W GONBE2.C

小文字だと，正しくコンパイルされませんのでご注意ください。なお，プログラムにはバグはありません。

バグに関するお問い合わせは
☎03(230)7683(直通)
月～金曜日16：00～18：00

お問い合わせは原則として，本誌のバグ情報のみに限らせていただきます。入力法，操作法などはマニュアルをよくお読みください。

また，よくアドベンチャーゲームの解答を求めるお電話をいただきますが，本誌ではいっさいお答えできません。ご了承ください。

創刊8周年 次は100号記念だ どうしよう！

▼Oh！X初のディスク付録“創刊8周年記念PRO-68K”はいかがでしたか？ 1.2Mバイトのフロッピーディスクにこれだけの内容を詰め込んだ付録をつけるのは今後もほとんどないのではと自負しています（ちなみにダンプリストにすれば本誌1200ページ分ものオブジェクトになる）。また，リスト打ち込み許容範囲を超えるツール群を皆さんに用意してもらうことにより，今後はそれらを利用した記事も安心して出せることになります。PurePASCALの連載も始まりました。OPMD.Xを使ったミュージックプログラムの投稿なども期待したいところです。

▼2月号のアンケートではご協力ありがとうございました。ここでプレゼント発表です。抽選の結果，シャープの液晶テレビ「クリスタルトロン」は京都府にお住まいの高田智之さんが当選されました。おめでとうございました。また，そのほか100名の方に記念品としてOh！Xのロゴ入りビニールファイルをお送りします。

▼ご回答いただいた結果，予想以上に面白いデータが得られました。記事のなかでも指摘されているとおり，これが本誌読者の全体像ではないことはおわかりでしょう。X68000ユーザーの場合，約半数がCコンパイラを持ち，4割近い人が58,000円のZ'sSTAFFを持ち，さらに2割以上がサイバースティックを持つというのですから，あのアンケートに答えられた人はかなりのユーザー層に違いありません。彼らがOh！Xの読者層の核をなすことができる人々とすれば心強いものがありますね。

▼Oh！Xでは引き続き筆者および協力スタッフを募集しております。応募資格は東京近郊にお住まいの社会人および学生でOh！Xの誌面作りに参加したい人。希望者は，住所・氏名・年齢・電話番号を明記のうえ，自己PR（投稿経験があればそれも）などに加え，自由原稿を6000字以内（本誌約2ページ分）にまとめ，Oh！X編集部「スタッフ希望」係までお送りください。お待ちしています。

投稿応募要領

- 原稿には，住所・氏名・年齢・職業・連絡先電話番号・機種・使用言語・必要な周辺機器・マイコン歴を明記してください。
- プログラムを投稿される方は，詳しい内容の説明，利用法，できればフローチャート，変数表，メモリマップ（マシン語の場合）に，参考文献を明記し，プログラムをセーブしたテープ（ディスケット）を添えてお送りください。また，掲載にあたっては，編集上の都合により加筆修正させていただくことがありますのでご了承ください。
- ハードの製作などを投稿される方は，詳しい内容の説明のほかに回路図，部品表，できれば実体配線図も添えてください。編集室で検討の上，製作したハードが必要な場合はご連絡いたします。
- 投稿者のモラルとして，他誌との二重投稿，他機種用プログラムを単に移植したものは固くお断りいたします。

あて先

〒102 東京都千代田区九段南2-3-26井関ビル
日本ソフトバンク出版部
Oh!X「㋢㋒㋮㋯」係

SHIFT・BREAK

▶緑水Softwareの皆さんへ。冬コミで買ったPoker Sistersのディスクがアンフォーマットでした。連絡先がわからないので連絡ください。スタッフ一同，遊べるようになる日を心待ちにしております。しかし，コミケのカタログを探すのにこんなにかかるたぁ思わんかったい。ちゃんと連絡先を書いておいてほしかったな。（H.U.）

▶今回付録ディスクに収録された「INTEGRAL X」はいかがだったでしょう。X1CからX1turboZに至る私のX1魂が伝わったでしょうか？ フォーマットやディスクコピーなど，X68000でやるまでもないことをX1にまかせてしまう。などなど，結構面白いこともできそうなので，皆さまのX1の座右のソフトにしてください。（亀）

▶やっっっとX68000を買いました。これで自分のマシンでドラスピができます（とっくにクリアしてるけど）。非68ユーザーの皆さん，お気に入りのソフトを買って励みにしましょう。ビデオのときもおんなじパターンだったしなぁ。そーいえばLDのソフトも持ってんだよなぁ，ハードはないけど。免許があるから車もそのうち……。（S.K.）

▶今はまだ4月。新製品の季節である。バイクで走っていても，発売直後の新車が街にあふれ，雨後のタケノコのよう。毎年のモデルチェンジに追従できる裕福な人が多いものだ。一方，モデルチェンジしても損した気分にならないマシンも確かにある。そういうマシンと長く付き合っていきたい。バイクにしてもパーソナルコンピュータにしても。（A.T.）

▶小説や漫画を読んでいるとき，頭の中で無意識のうちにキャラクターに応じた“声”を使い分けたりする，よね。TV化されたときに違和感があったりするのは，その頭の中の声とのギャップのせいなんだと思う。ところがしばらくして読み返してみると，いつの間にかTV版の声とイントネーションで読んでいる。あれって，なんか，悔しい。（Mu）

▶そこで，三上晴子の個展に行ってきた。トーヨコ地球環境研究所という怪しいところで無菌なクリーンルームも使ったりした“Information Weapon”。ジャンクの基板を貼り合わせて光らせたりモノクロCRTを組み合わせたりしてなかなか面白かった。ボクらも情報やフォン・ノイマンの亡霊に負けないよー頑張ろう（亡霊はお祓いだ）。（K）

▶ドラクエに興じているうちに桜も終わり，送別会と歓迎会の多さが別れと出会いの季節の到来を感じさせてくれる。喜びか溜め息か，春はとってもスリリング。今年の花見はひとりでやったが来年こそはたぶん……やはりひとりかなあ。いずれにせよ何かを期待させる季節，それが春なのさ。う～ん，意味不明。（新作のウルトラQを観て後悔したKO）

▶「Confusion will be my epitaph……」，これがOh！X編集部に配属されて2週間たった感想だった…。というわけで，今月号から当編集部にお世話になることになった新入社員Aです。どうぞよろしくお願いします。私はパソコン歴10年，というと聞こえはよろしいが実はここ5年ほどは触っていないのでわからないことだらけなのです。（A）

▶先日，久しぶりにテレビで松本伊代を見かけた。まだ生きていたのかと思いつつ番組をよくよく見ると，なんと伊代ちゃんがピンクレディ（！）の振りまねをさせられているではないか。しかもカラオケもなくえんえんと，である。時の流れはあな恐ろし。そういえば彼女ももう25か，と思ったとき自分も同じ年なのに気づいた。ひゅうぅ～。（E.O.）

▶青天の霹靂とはこういうことをいうのだろうか。父が逝った，ガンだった。兄も死んだ，エイズで…。そのショックで母も崩れた。というわけで残された年老いた姉（108歳）の面倒を見るために故郷パプアニューギニアに戻らなければならなくなった。今後は一読者として，遙か異国の地からOh！Xを見守っていきたいと思う。では，また逢う日まで。（S）

▶ささやかな（しかし地上最強の）オマケのはずだったが，風雲急を呼び嵐をついての6月号となった。「毎月やれ」といわれても胸を張って「できません」といえるくらいの内容にはなったと思う。オマケなど，およそOh!Xらしからぬことでもある。もちろん，もっと違うかたちのモノも考えた。でも，まあ，やっぱり，こんなもんじゃないか？（U）

▶ウイルス問題が多くの人を混乱させた（人騒がせな通報や，誤報もあったし）。本誌では社会的影響も考え，できるだけ慎重に対処してきたつもりだが，あと2カ月ワクチンの配布が早ければこんな騒ぎは防げたかもと思うと辛い。読者の皆さんの理解と協力をお願いしたい。さて，S氏が去り，新卒のA君が入社した。気持ちを新たに頑張りたい。（T）

microOdyssey

人々が自由に海外旅行を楽しむようになってから25年が過ぎようとしている。それにつれて生活自体も豊かなものになってきている。もうこれ以上豊かになる必要はあるのだろうか，と思うほどである。とはいえ，人の欲というものはキリがないらしく，もっと豊かな，もっと向上した生活を，と誰もが考えているものだ。

その豊かな生活の延長線上にレジャー施設がある。いまや日本国内のみならず，世界各国に日本の大手企業がレジャー施設を作っている。時間に追われる人々がひとときの幸せに浸り，ストレスを発散させるためだという。だが，それは本当の意味で人のためのものなのだろうか。

レジャー施設の建設，それは自然破壊にほかならないのである。私が思うに，自然は自然なままがいちばん美しい。砂漠や南極，人の手の入れられていないところは美しく，自然の恵みと支配に満ちている。チベットやアンデスに住む人々は，私たちのような文化的な生活こそしていないが，ストレスを溜めることもなく，幸せそうな笑顔を向けている。彼らを見ているとなにも自然を壊してまで楽しもうとしなくても，まずストレスを溜めるような生活様式を変えるほうが先ではないか，と思うのである。

レジャー施設の建設以外でも，近年さまざまな面で環境破壊が問題になっている。身近な問題としては，フロンガスによるオゾン層破壊がある。最近，渋谷などにいる若者たちは流行りの服装に身を包み，髪の毛を男であれ女であれしっかりとセットしている。が，あのセット剤がくせものである。フロンガスを使用，まさに自然破壊である。地球環境の危機に対する関心の高まりを表すといわれるアースカラーを身にまとい，フロンガスを頭に振りまく姿はまさに滑稽である。もっとも，日焼けがおしゃれという彼らにとっては，紫外線が多くなることもありがたいのかもしれないが。

そして，核実験や原子炉事故などによる大気汚染の数々。なんのための核実験かは知らないが，放射線を撒き散らし土に染み込ませ，その大地で育てられた作物や家畜もまた放射能をいっぱいあびているのである。どだい核兵器の使用はしない，といいながら核実験をしているのだから，わけがわからない。

人々のためにあるといわれる人工的なレジャー施設の建設，おしゃれのためにと使っているフロンガス，そして核実験，それらがどれだけこの地球を傷つけ苦しめていることか。ひいてはそれが地球全体を破滅に導くことになるのである。自然破壊は酸素を減らし，オゾン層の破壊は皮膚ガンで苦しむ人を増やしていく。核実験は放射性発ガンのもととなる。おかげで最近の地球は妙な気候になり，地震も増えていると聞く。自分で地震を引き起こすとは，さながらポピュラスの世界だ。人々は自分たちの首を自分たちで絞めているとなぜ気がつかないのだろうか。それとも知っていてやっているのだろうか。もう，取り返しがつかないから，と。

私たちの地球は，以前の青かった姿とは似ても似つかない姿に変わりつつある。すべては自然破壊によるものだ。にもかかわらず，このうえ月までも観光地にしようとする計画さえある。地球のみならず，月までも汚染するつもりなのか，と思うと悲しくなってきた。　(E. O.)

1990年7月号6月18日(月)発売

特集　マシン語への第一歩
ノーマルX1対応INTEGRAL X1
全機種共通システム
リロケータブルアセンブラ&リンカ　WZD/WLK
Oh!X LIVE in '90
T-SQUARE OMENS OF LOVE/夢幻戦士ヴァリスII Sacred Sacrifice 他
連載
PurePASCAL入門/ハードウェア工作入門/X68000マシン語プログラミング 他

バックナンバー常備店

東京	神保町	三省堂神田本店5F	03(233)3312
	〃	書泉ブックマートBI	03(294)0011
	〃	書泉グランデ5F	03(295)0011
	秋葉原	T-ZONE 7Fブックゾーン	03(257)2660
	八重洲	八重洲ブックセンター3F	03(281)1811
	新宿	紀伊国屋書店本店	03(354)0131
	高田馬場	未来堂書店	03(200)9185
	渋谷	大盛堂書店	03(463)0511
	池袋	リブロ池袋店	03(981)0111
	〃	西武百貨店9F コンピュータ・フォーラム	03(981)0111
神奈川	横浜	有隣堂横浜駅西口店	045(311)6265
	〃	有隣堂ルミネ店	045(453)0811
	藤沢	有隣堂藤沢店	0466(26)1411
神奈川	厚木	有隣堂厚木店	0462(23)4111
	平塚	文教堂四の宮店	0463(54)2880
千葉	柏	新星堂カルチェ5	0471(64)8551
	船橋	リブロ船橋店	0474(25)0111
	〃	芳林堂書店津田沼店	0474(78)3737
	千葉	多田屋千葉セントラルプラザ店	0472(24)1333
埼玉	川越	黒田書店	0492(25)3138
	川口	岩渕書店	0482(52)2190
茨城	水戸	川又書店駅前店	0292(31)0102
大阪	北区	旭屋書店本店	06(313)1191
	都島区	駸々堂京橋店	06(353)2413
京都	中京区	オーム社書店	075(221)0280
愛知	名古屋	三省堂名古屋店	052(562)0077
	〃	パソコンΣ上前津店	052(251)8334
	刈谷	三洋堂書店刈谷店	0566(24)1134
長野	飯田	平安堂飯田店	0265(24)4545
北海道	室蘭	室蘭工業大学生協	0143(44)6060

定期購読のお知らせ

Oh!Xの定期購読をご希望の方は，とじ込みの振替用紙の「申込書」欄に何年何月号からをご記入のうえ，**年間購読料6,720円**(税込)を添えてお申し込みください。その際，裏面の通信欄に「○年○月号よりOh!X定期購読希望」と忘れずに明記してください。なお，すでに定期購読をご利用いただいている方には，購読期限終了と同時にご通知申し上げますので，同封の払込用紙をご利用ください。

海外送付ご希望の方へ

本誌の海外発送代理店，日本IPS(株)にお申し込みください。なお，購読料金は郵送方法，地域によって異なりますので，下記宛必ずお問い合わせください。

日本IPS株式会社
〒101 東京都千代田区飯田橋3-11-6
☎03(238)0700

Oh!X 6月号
■1990年6月1日発行　特別定価780円(本体757円)
■発行人　孫　正義
■編集人　橋本五郎
■発売元　(株)日本ソフトバンク
■出版事業部　〒102 東京都千代田区九段南2-3-26　井関ビル
Oh!X編集部　☎03(230)7681
出版営業部　☎03(230)7670 FAX 03(262)8397
広告センター　☎03(297)0181
■印　刷　凸版印刷株式会社

信頼と実績のお店
BASIC HOUSE

今がチャンス

X68000を御買上げの方にもれなく
下記X68000グッズのいずれか1つを
プレゼント!

A. PROSTAFF ジャンパー
B. X68000目覚し時計
C. ツタンカーメンZIPPO
D. ビジネスバッグ

サポート万全! 我々にお任せください!

NEW X68000 SUPER-HD

特別価格にて予約受付中!

- 大容量80MB 3.5′HD内蔵
- SCSIインターフェイス標準装備
- 疑似マルチタスク、マルチウィンドウを実現した"SX WINDOW"を搭載
- 処理速度大幅向上(平均2倍)

BASIC HOUSE超特価
(限定品)

CZ-602C-GY+CZ-612D-GY
¥3□□,000
CZ-652C-GY+CZ-612D-GY
¥3□□,000

NEW X68000 EXPERT II

CZ-613C
CZ-612D
CZ-8PC4
定価¥667,600
BASIC HOUSE特価

NEW X68000 PRO II

CZ-663C
CZ-605D
CZ-8PC3
定価¥609,800
BASIC HOUSE特価

AMIGA

CALL

Macintosh

CALL

周 辺 機 器

品名	価格
CZ-8PC3	BH特価
CZ-8PC4	¥ 99,800
CR-3415CL	¥109,800
CR-3410CL	¥ 98,000
VP-2050	BH特価
CZ-8NS1	¥188,000
GT-6000	¥178,000
GT-4000	¥198,000
HS-10R II	¥ 49,800
HXD040	¥118,000
HXD042	¥128,000
IT-X640	¥158,000
IT-X680	¥188,000
MD24FS5	BH特価
MD12FS	BH特価
XE-1 PRO	BH特価
CYBER STICK	¥ 23,800
CZ-6BF1A	¥ 38,000
CZ-6BG1	¥ 59,800
CZ-6BM1	¥ 26,800
SX-68M	¥ 19,800
C Compiler PRO-68k	¥ 39,800
Mu-1	¥ 19,800
マジックパレット	¥ 19,800
Zs STAFF PRO-68k	¥ 58,000
C-TRACE68	¥ 68,000
CARD PRO68k	¥ 29,800
CZ-6EB1	¥ 88,000
CZ-8NT1	¥ 13,800
AN-S100	¥ 36,600

全国どこでも発送可　長期クレジットOK　送料全国均一¥1,000　宅配便にて即日配送

株式会社計測技研
本社営業部/マイコンショップ/通販部　宇都宮市竹林町503−1　TEL0286-22-9811　FAX0286-25-3970
大田原営業所/マイコンショップ　大田原市美原1−13−4　TEL0287-23-5352　FAX0286-23-5364
マイコンショップ BASIC HOUSE お申し込み・お問い合せは ☎0286-22-9811(代)

AVCフタバ

☎03(253)7661

〒101 東京都千代田区外神田3-2-3 ☎03-253-7611(代)

今すぐ もよりの電話から	仙台 022-264-3704	名古屋 052-452-3271	広島 082-295-6873
札幌 011-611-5104	新潟 0252-75-4175	大阪 06-311-3931	福岡 092-481-2494

高価下取り、買取りいたします、お問合せ下さい。

AVC本店　AVCジャンプ　AVCホップ　至新宿　電波ビル　外堀通り　リビナ　日通ビル　至上野　ヤマギワ　中央通り　至神田　秋葉原駅

Welcome! ご来店もどうぞ。

X68000の情報のすべて!(当店はX68000の認定代理店です。お気軽にご相談下さい)

X68000 待望の新しい仲間登場!!

X68000 PERSONAL WORKSTATION EXPERTII・EXPERTII HD

EXPERTII・EXPERTII HD
集積度を高めた"マンハッタンシェイプ"3Mの大容量メモリを搭載。本格的なウインドウシステム、SX-WINDOW搭載。

〔写真のモニタは別売です。〕

CZ-603C 標準価格¥338,000
CZ-613C 標準価格¥448,000
AVC特価

X68000 PERSONAL WORKSTATION PROII・PROII HD

PROII・PROII HD
拡張I/Oポートを4スロットを搭載し、汎用性と低価格が魅力。
もちろん、SX-WINDOW搭載。

CZ-653C 標準価格¥285,000
CZ-663C 標準価格¥395,000
AVC特価

X68000

在庫稀少価格はお電話で!
CZ-602C
CZ-612C
CZ-652C
CZ-662C

お勧めディスプレイコーナー　組合せは自由、価格はお気軽にご相談下さい。

CZ-604D 標準価格¥94,800 AVC特価
●0.31mmドットピッチ ●2モードオートスキャン ●ステレオスピーカ搭載 ●チルト台同梱

CZ-613D 標準価格¥135,000 AVC特価
●ドットピッチ0.31mm ●TVチューナー搭載 ●ステレオスピーカー搭載 ●チルト台同梱

CZ-603D 標準価格¥84,800 AVC特価
●0.31mmドットピッチ ●TVチューナ無し ●3モードオートスキャン ●チルト台同梱

CU-21HD 標準価格¥148,000 AVC特価
●0.52mmドットピッチ ●21型ディスプレイ ●3モードオートスキャン ●ステレオスピーカ搭載

CZ-605D 標準価格¥115,000 AVC特価
●ドットピッチ0.39mm ●TVチューナー搭載 ●ステレオスピーカー搭載 ●チルト台同梱

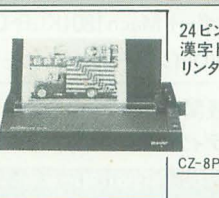
CZ-602D 標準価格¥99,800 AVC特価
●ドットピッチ0.39mm ●TVチューナー搭載 ●チルト台同梱

型番	品名	標準価格	販売価格
CZ-6TU	システムチューナー	¥33,100	AVCフタバ特価
BF-68PRO	CRTフィルター	¥19,800	AVCフタバ特価
CZ-8NS1	カラースキャナー	¥188,000	AVCフタバ特価
CZ-6BN1	スキャナー用パラレルボード	¥29,800	AVCフタバ特価
CZ-6VT1	カラーイメージユニット	¥69,800	AVCフタバ特価
CZ-8BV2	カラーイメージボード	¥39,800	AVCフタバ特価
CZ-8BR1	立体映像セット	¥29,800	AVCフタバ特価
CZ-8DT2	パーソナルテロッパ	¥44,800	AVCフタバ特価
CZ-8BS1	FM音源ボード	¥23,800	AVCフタバ特価
CZ-8NJ1	ジョイカード	¥1,700	AVCフタバ特価
CZ-8NM2A	マウス	¥6,800	AVCフタバ特価
CZ-8NM3	マウス・トラックボール	¥9,800	AVCフタバ特価
CZ-6SD1	システムラック	¥44,800	AVCフタバ特価
AN-S100	アンプ内蔵スピーカー	¥36,600	AVCフタバ特価
CZ-6EB1	拡張I/Oボックス	¥88,000	AVCフタバ特価
CZ-8PG1	24ピンカラープリンター(80桁)	¥130,000	AVCフタバ特価
CZ-8PK10	24ピンプリンター(136桁)	¥97,800	AVCフタバ特価
IO-735X	カラージェットプリンター	¥248,000	AVCフタバ特価
CZ-6BE1A	1M増設RAMボード	¥38,000	AVCフタバ特価
CZ-6BE2	2M増設RAMボード	¥79,800	AVCフタバ特価
CZ-6BE4	4M増設RAMボード	¥138,000	AVCフタバ特価
CZ-6BP1	数値演算プロセッサー	¥79,800	AVCフタバ特価
CZ-6BC1	FAXボード	¥79,800	AVCフタバ特価
CZ-6BM1	MIDIボード	¥26,800	AVCフタバ特価
CZ-6BU1	I/Oボード	¥39,800	AVCフタバ特価
CZ-6BL1	LANボード	¥268,000	AVCフタバ特価
CZ-243BS	サイバーノート	¥19,800	AVCフタバ特価
CZ-240BS	スティショナリー	¥14,800	AVCフタバ特価
CZ-223CS	通信ソフト	¥19,800	AVCフタバ特価
ゲームソフト		20% OFF	
CZ-8TM2	モデムユニット	¥49,800	AVCフタバ特価
CZ-252MS	Musicstudio	¥28,800	AVCフタバ特価
CZ-247MS	MUSIC(MID)	¥28,800	AVCフタバ特価
CZ-221HS	NEW Print Shop	¥19,800	AVCフタバ特価
CZ-228BS	TOP給与計算エキスパート	¥200,000	AVCフタバ特価
CZ-227BS	TOP財務会計	¥200,000	AVCフタバ特価
CZ-220BS	DATA	¥58,000	AVCフタバ特価
CZ-212BS	BUSINESS	¥68,000	AVCフタバ特価
CZ-219SS	OS-9	¥29,800	AVCフタバ特価
CZ-211LS	Ccompiler	¥39,800	AVCフタバ特価
CZ-234LS	AI-68K	¥188,000	AVCフタバ特価
CZ-620H	20MBハードディスク	¥178,000	AVCフタバ特価
CZ-64H	40MBハードディスク	¥120,000	AVCフタバ特価
LHD-34V	40MBハードディスク(ロジテック)	¥153,000	¥117,000
LHD-32V	20MBハードディスク(ロジテック)	¥128,000	¥98,000

CZ-8NJ2

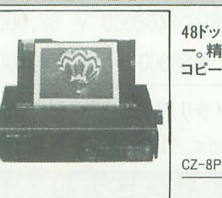
アナログジョイステック
標準価格 ¥23,800

AVC特価¥???

CZ-8PG2

24ピンカラー。
漢字ドットインパクトプリンタ

CZ-8PG2……¥160,000

AVC特価¥???

CZ-8PC4

48ドット熱転写プリンター。精密な文字、ハードコピーも可能。

CZ-8PC4……¥ 99,800

AVC特価 ¥64,800

CZ-8PC3

24ドット熱転写カラープリンター

標準価格……¥65,800

AVC特価 ¥39,800

●頭金なし(手軽な電話クレジット)●製品先取り(お支払いは約1～2ヶ月後から)●低金利クレジット(1回の支払いは2,700円以上で3～48回。ボーナス併用も可)●カレッジクレジット(保証人なし。但し満20歳以上の学生の方)●18歳未満の方(ご両親が代理購入者としてお申し込み下さい)
●納期(通常の場合、当社に申込書が到着後1週間以内。特に人気のある商品で品薄の場合、少々納期が遅れることがありますので御了承下さい)
●完全保証(すべてメーカー保証書付。アフターケア万全)●全国代引(お届けした者に、代金をお支払いいただく方法です。但し手数料1,000円)

AM10時からPM7時まで受付 日曜・祝日も営業

●但し消費税(3%)は別途請求させていただきます。●分割回数は3回～48回まで自由に選べます。

●セットの組合せは自由!広告に出ていない他の機種はお問合せ下さい

X68000 新型発売記念 特価セール

X68000 EXPERT/PRO

New X68000新発売!

機種	定価
CZ-603C	¥338,000
CZ-613C	¥448,000
CZ-623C	¥498,000
CZ-653C	¥285,000
CZ-663C	¥395,000
〈ディスプレイ〉	
CZ-603D	¥84,800
CZ-604D	¥94,800
CZ-605D	¥115,000
CZ-613D	¥135,000

CZ-602C-GY(本体) ¥356,000
CZ-603D-GY(ディスプレイ) ¥84,800
定価合計¥440,800
(このセットに限り、送料+消費税込)
'90年6月末迄 ズバリ大特価!¥305,000

CZ-653C(本体) ¥285,000
CZ-601D(ディスプレイ) ¥110,700
定価合計¥395,700
春の見体験フェア 展示品特別価格 ¥288,000
(このセットに限り、送料+消費税込)

※代金は商品引換着払いでもOKです。

●CZ-602、603単体でも大特価!お問い合わせください。

MZ2500下取り/MZ2500からMZ2861(定価¥328,000)に買い替え下取後 特価¥165,000
CZ600C下取り/CZ600CからCZ623(X68000 SUPER)に買い替え下取後 特価¥300,000

AIBIT アイビット電子株式会社

FM TOWNS お買い得セット

1.TOWNS-1	¥338,000
2.FMT-ME(1M)	¥60,000
3.FMD-FD301	¥28,000
4.FMT-KB101	¥20,000
5.FMT-DP531	¥89,800
6.TOWNS-OS V1.1 L20	¥20,000
定価合計	¥555,800

大特価! ¥285,000

ハガキもOK、New MZプリンタ

漢字カラー熱転写プリンタ シャープMZ-1P22

好評発売中!

〈24×24ドット漢字●7色カラー●漢字30字/秒高速印字●MZ1P17とフルコンパチ●5KBのバッハメモリ付〉適応パソコン:MZ2000、2500、5500、6500シリーズ、X1シリーズ、X68000シリーズ他。

標準価格¥59,800➡特価¥25,000

パソコンファクスMZ-1V01

"プリンタ・コピー・ファクス"1台3役のスグレモノ 限定セット販売!

●MZ25セット(インターフェースソフト付)
標準価格合計¥342,800➡¥168,000
●MZ-1V01(本体のみ)
標準価格合計¥278,000➡¥98,000

シャープMZ-1X30 モデムホン

(1×19上位機種)

〈300/1200BPS全2重通信対応モデム内蔵●音声入出力端子付●ダイヤルパルス/プッシュボタン対応●プッシュボタン音解析機能●シャープ手順、CCITT、V25bis通信手順サポート〉

標準価格¥98,000➡大特価

パソコンと専用ワープロをひとつにした16ビット シャープMZ-2861

ワープロソフト「書院28」
MS-DOS V3.1 装備
エミュレーションソフト搭載

定価¥328,000➡
大特価¥198,000
(ディスプレイは別)

MZ-2861用ソフト(UPシリーズ)

●1P-1251(デスクアップ)……定価¥88,000➡特価¥20,000
●1P-1253(クリッパー)……定価¥77,000➡特価¥20,000
●1P-1254(プランナップ)……定価¥88,000➡特価¥20,000

シャープMZ-2520

定価¥159,800➡大特価¥80,000

新発売!《限定発売》

'89プログラム大賞グランプリ受賞作「HEAVY METAL」搭載

PC-E500PJ

定価¥28,800
➡大特価

●ご購入の方にもれなく「ポケコンジャーナル特別号」を進呈。

PC-500と各種パソコンをつなぐインターフェースケーブル
CE-140T ¥8,800

拡張機器他

●シャープCZ6BM1(X68用MIDIボード)¥26,800⇒¥23,000
●シャープCZ-8GR(X1.GRAM)‥¥32,000⇒¥12,000
●シャープCZ-8EB3(I/Oボックス)‥¥33,800⇒¥28,000
●シャープCZ-8BK3‥‥(X1)‥‥¥13,800⇒¥11,700
●シャープCZ-8BK4‥‥(X1)‥‥‥¥6,800⇒¥5,700
●シャープCZ-8BGR2・(X1)‥‥‥¥14,800⇒¥4,000
●シャープCZ-8BS1‥‥(X1)‥‥¥23,800⇒¥19,500
●シャープCZ-64H(ハードディスク)〈CZ-602・652内蔵用〉………特価
●シャープCZ-8NJ2(インテリジェントコントローラー)¥23,800⇒¥18,500
●シャープCZ-81Tチルトスタンド‥‥‥¥8,500⇒¥1,000
●シャープMZ-1U08(1500,700拡張)‥¥25,000⇒¥12,000
●シャープMZ-1U03(1500,700拡張)‥¥35,000⇒¥15,000
●シャープMZ-1X22モデムユニット‥¥21,800⇒¥13,000
●シャープMZ-1R12 RAM………¥35,000⇒¥8,000
●シャープMZ-1E29 (MZ)……¥17,800⇒¥9,800
●シャープMZ-1E30(2500ハードディスクインターフェイス)……¥25,000⇒¥22,500
●シャープMZ1E26(ボイスコミュニケーション)…¥24,800⇒¥13,000
●シャープMZ-1U09…(2500)……¥9,000⇒¥7,200
●シャープMZ-1M03…(5500)・¥69,000⇒¥35,000
●シャープMZ8BC04‥(2000)‥¥18,000⇒¥8,000
●シャープMZ-8B104‥(2000)・¥45,000⇒¥18,000
●シャープMZ-1R11……(5500)・¥80,000⇒¥30,000
●シャープMZ-1R24…(1500)‥¥22,000⇒¥6,000
●シャープMZ-1R26A‥(2500)・¥13,000⇒¥12,800
●シャープMZ-1R27A‥(2500)・¥13,000⇒¥10,000
●シャープMZ-1R28A・(2500) ¥13,000⇒¥10,000
●シャープMZ-1R29A‥(2500)・¥32,000⇒¥10,000
●シャープMZ-1T02… (2200)‥¥19,800⇒¥8,500
●シャープMZ-1T03…(1500)‥¥12,000⇒¥8,500
●シャープMZ-1X29…………¥13,800⇒¥11,000
●テレシステムRM-25E(2500RAMファイル)¥428,800⇒¥38,500
●シャープ CZ-6SP1代品チルト台……特価¥3,500
●シャープCZ-8BE2X1 320KRAM……¥29,800⇒¥25,300
●シャープCZ-8BM2X1 232Cマウスボード‥¥19,800⇒¥16,800
●シャープX1、MZ用マウス…………特価¥4,800
●シャープX1用ジョイカード………………¥1,500
●富士通16βキーボード(親指)・¥25,000⇒¥20,000
●シャープMZ-3500キーボード………… ¥8,000
●シャープMZ-5500キーボード………… ¥8,000
●シャープ2000/2200キーボード ……… ¥8,000
●シャープMZ-1E08…………¥9,000⇒¥8,000
●シャープMZ-1M08…………¥10,000⇒¥6,500
●シャープCZ-6BE1(X68000増設RAM)……¥35,000⇒¥29,500
●シャープCZ-6BE1A(X68000増設RAM)……¥38,000⇒¥23,800
●アイテックPIO-6BE1-A(IOデータメモリー拡張ボード)¥25,000⇒¥21,500
●アイテックPIO-6BE2-2M(IOデータメモリー拡張ボード)¥50,000⇒¥42,500
●アイテックPIO-6BE4-4M(IOデータメモリー拡張ボード)¥88,000⇒¥74,500

(MZ-2861)

●シャープMZ1R35(1MB増設メモリーボード)‥¥55,000⇒¥19,000
●シャープMZ1R36(MZ-2861増設1MRAM) ¥45,000⇒¥15,000
●シャープSS-SC28M(SCSIインターフェイス)¥49,800⇒¥10,000
●シャープ1E39(RE232C 2CHボード)……¥39,800⇒¥13,000

プリンター

●シャープCZ-8PC3…………¥65,800⇒大特価
●シャープCZ-8PC4(黒・グレー)・¥99,800⇒大特価
●シャープMZ-1P27………¥268,000⇒¥214,400
●シャープMZ-1P28………¥148,000⇒¥118,400
●シャープMZ-1P29………¥168,000⇒¥134,400

フロッピーディスク

●シャープCZ501H(X68拡張用10M)・¥258,000⇒¥60,000
●シャープCZ-503F…………¥49,800⇒¥30,000
●シャープCZ-502F…………¥99,800⇒¥60,000
●シャープCZ-53F……………¥19,800⇒¥9,800
●シャープCZ-300F(CZ-3PCM付)…………¥13,000
●シャープCZ8PG1………¥130,000⇒¥100,000
●シャープCZ8PG2………¥160,000⇒¥130,000

ハードディスク

●アイテックIT-X640…………¥158,000⇒¥128,000
●アイテックIT-X68 …………¥198,000⇒¥158,000

ディスプレイ

●富士通FMTV-153…………¥108,000⇒¥76,000
●シャープMZ-1D27…………¥120,000⇒¥79,800

ソフト

(X68000用)

●CZ-230AS ニュージーランド‥‥¥8,800⇒¥7,040
●CZ-2310AS FULL.THRTTLE……¥8,800⇒¥7,040
●CZ-233AS PACMANIA………¥7,800⇒¥6,250
●CZ-222AS ARKANOID………¥7,800⇒¥6,250
●POPULOUS…………………¥9,800⇒¥7,850
●CZ-239AS THUNDARBLADE‥¥9,500⇒¥8,000

(MZ-2500用)

●1P-1215 COBOL…………¥13,800⇒¥11,700
●1P-1217 PROLOG…………¥11,300⇒¥11,700
●MZ-6Z001 2500 PCPM……¥16,800⇒¥14,200
●DANGER BOX……………¥5,800⇒¥2,000
●EXTRA HYPER DISK MONITOR……¥10,000⇒¥8,500
●EXTRA HYPER DISK MONITOR+…¥14,000⇒¥12,000
●FILE UTILITY<UT-25F>…………¥6,800⇒¥6,000
●FREE CALL…………………¥6,800⇒¥1,000
●G-EDIT2500………………¥8,000⇒¥7,000
●H.S-コントローラー…………¥9,600⇒¥8,500
●HuCAL日本語……………¥45,000⇒¥15,000
●エキサイトバイク…………¥6,800⇒¥2,000
●カレイドスコープ…………¥9,800⇒¥3,000
●カレイドスコープ2…………¥5,800⇒¥1,000
●ザ・ブラックオニキス…………¥7,800⇒¥3,000
●スーパー修理屋さん………¥12,000⇒¥10,200
●ムーンチャイルド…………¥7,800⇒¥3,000
●英雄伝説サーガ…………¥9,800⇒¥2,000
●五目並べ…………………¥4,800⇒¥2,000
●探検隊第2弾………………¥7,800⇒¥2,000
●プリントSHOP……………¥9,800⇒¥8,500
●プリントSHOPライブラリー1……¥4,500⇒¥3,800
●プリントSHOPライブラリー2……¥4,500⇒¥3,800

(X1用)

●日本語ワープロ将軍X1t……¥34,800⇒¥29,000
●日本語ワープロ侍 X1t………¥19,800⇒¥16,800
●CZ-8WB51 X1tディスクBASIC………¥9,800⇒¥3,500
●3CP/M X1 3" CPM…………¥16,800⇒¥5,000
●CZ-8BK3 X1t第二水準ROM‥¥13,800⇒¥11,700
●CZ-128SF X1.CP/M………¥13,800⇒¥11,700
●CZ-115LF X1 FORTRAN……………………品切
●CZ-116LF X1.C……………¥13,800⇒¥11,700
●CZ-117SF X1t LOGO………¥18,800⇒¥13,200
●CZ-118LF X1.COBOL………¥13,800⇒¥11,700
●CZ-126LF X1 APL…………¥13,800⇒¥11,700
●CZ-130SF X1t CP/M………¥14,800⇒¥12,500
●CZ-137SF X1t ZSSTAFF……¥19,800⇒¥16,800
●CZ-138SF X1 ZSSTAFF……¥13,800⇒¥11,700

(MZ-5500、6500SOFT)

●MZ-2Z013(MZ-5500MSDOS)
●MZ-2Z014(MZ-5500TODAY)
●MZ-2Z023(MZ-5500GW.BASIC)
●MZ-2Z028(MZ-6500GW.BASIC)
●MZ-2Z025(MZ-5500ワープロ)
●MZ-2Z029(MZ-6500TODAY)

本体●シャープCZ-820、822、880、881、MZ-3500、2520、2861、2200、X68000、CZ-612、662、602、652 ●富士通FM-77AV-1、77AV-2、77AV20、77AV40 ●NEC PC-9801N ●東芝J3100SS

《全商品新品完全保証付》

■シャープポケコン全商品販売中。カタログ、特価表ご請求ください(〒72)。

アイビット推奨ディスプレイ

●三菱XC-1498CII
(14型アナログ)
ドットピッチ0.28
定価¥107,000⇨
特価¥59,800

XC-1498CII対応パソコン機種:PC-9801シリーズ/PC-286シリーズ/PC-386シリーズ/PC-8801シリーズ
(上記機種には付属の接続ケーブルで、接続可能)

●シャープCZ-830D・BK
(14型)
2モードオートスキャン方式
(アナログ/デジタル)
定価¥98,000⇨
特価¥54,800

CZ-830D対応パソコン機種:CZ880C/881C、X1/TURBOシリーズ。ケーブルは本体付属を使用。NEC PC-8801・9801シリーズ(XA・XLのみ不可)MZ700/1500/2000/2200/2500各シリーズ(推奨品シャープ8D8K)。

●シャープCZ-611D-GY
(15型アナログTV/3モードオートスキャン)
定価¥145,000⇨
特価¥89,800

CZ-611D対応パソコン機種:※X1シリーズ/※X1 turboシリーズ/X1 yurboZシリーズ/X68000シリーズ/PC8801シリーズ/PC-9801シリーズ/PC-286シリーズ
(※は接続ケーブルAN1506が必要です)

●シャープCZ-602D-GY・BK
(15型カラーディスプレイTV)
ドットピッチ3.9
定価¥98,000⇨
特価¥79,000

CZ-602D対応パソコン機種:※X1シリーズ/※X1 turboシリーズ/X1 yurboZシリーズ/X68000シリーズ/PC8801シリーズ/PC-9801シリーズ/PC-286シリーズ
(※は接続ケーブルAN1506が必要です)

☎0426-45-3001~3
FAX.0426-44-6002
●営業時間/10:00~19:00●電話受付/20:00迄可●定休日/日曜日(祭日営業)
SHARP SUPER XEX SHOP
アイビット電子株式会社 〒192 東京都八王子市北野町560-5

上記の広告商品はすべて店頭販売もしております。

全国通販 信売
北海道から沖縄まで

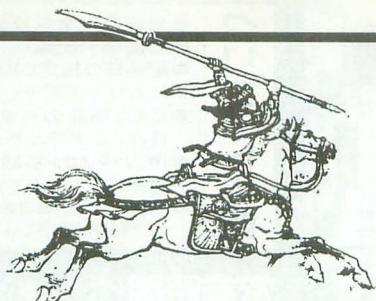
★送料はご注文の際にお問い合わせ下さい。
★掲載の商品は、すべて新品、保証書付きです。
★掲載の商品は充分用意してありますが、ご注文の際は、在庫の確認の上、現金書留または、銀行振込でお申し込み下さい。全商品クレジットでも扱っております。
★お申し込みの際は必ず電話番号を明記して下さい。
★商品、品切れの節はご了承下さい。

富士銀行八王子支店 (普)1752505

●本誌発売時には上記価格よりさらにお求めやすい価格に変更されている場合があります。●一部を除き上記商品価格には消費税は含まれておりません。全ての商品に対し別途3%の消費税がかかりますのでご了承ください。

KOEI

三國志

この度は三國志「光栄特別試写会」に

光栄特別試写会

多数のご応募をいただきましてありがとうございました。

総額1,500万円

厳正なる抽選の結果、右記の方々がご当選となりました。

豪華プレゼント

おめでとうございます。

当選者発表。

中国「三國志」の旅 ……カップルで20名様

香川県高松市 田村節雄様　北海道札幌市 梶原賢太様
宮城県栗原郡 佐藤　源様　東京都板橋区 川嶋智子様
神奈川県横浜市 藤井知子様　大阪府豊中市 占部勝洋様
東京都小平市 東島直久様　新潟県柏崎市 宮嶋政敏様
大阪府東大阪市 益井鉄男様　東京都江戸川区 川俣光明様

日本電気㈱PC-98DOセット ……8名様

千葉県船橋市 永田和弘様　愛媛県新居浜市 上野　剛様
東京都豊島区 松本幸一様　東京都中野区 川島秀樹様
宮城県宮城郡 佐藤美由紀様　東京都新宿区 柳澤　学様
愛知県小牧市 伊藤公晴様　北海道登別市 吉田文敏様

富士通㈱FM-TOWNSセット ……8名様

千葉県市原市 若月義治様　兵庫県神戸市 鈴木建次様
兵庫県神戸市 武智勇二様　宮崎県宮崎市 永野武雄様
北海道札幌市 三部　剛様　静岡県富士宮市 加藤正人様
神奈川県平塚市 高瀬　達様　兵庫県養父郡 辻垣晃一様

松下電器産業㈱A1 WSXセット ……8名様

京都府京都市 米岡　亮様　兵庫県神戸市 工藤直樹様
愛知県海部郡 牛田照美様　千葉県千葉市 池田義満様
神奈川県藤沢市 関根清美様　静岡県静岡市 徳田憲史様
北海道札幌市 岸本裕章様　栃木県今市市 伊藤　洋様

◉下記当選者は賞品の発送をもって代えさせていただきます。
光栄「三國志」ファミコンゲームソフト………100名様
千葉県松戸市 中島勇人様 他99名様
光栄「三國志」サウンドウェア(CD)………100名様
北海道上川郡 成沢信良様 他99名様

●協賛/東宝東和㈱・日本電気㈱・富士通㈱・松下電器産業㈱
●協力/全日空㈱・㈱アスキー・㈱角川書店・㈱日本ソフトバンク・㈱電波新聞社・徳間書店インターメディア㈱〈テクノポリス編集部〉

株式会社 光栄　〒223 横浜市港北区日吉本町1-4-24 Tel.044-61-6861代

《広告の半ページ》 ぼやき続けて二周年。責任者でてこい！

月刊 電脳倶楽部 90年6月号(Vol.25) 5月18日発送

2HDディスクに入ったX68000のための雑誌だっ！

辞書・テキスト相互コンバータ
DICCONV. R

BEEP音をとっかえひっかえする
BELL. X

オリジナルアイコンデータ集

さらにさらに
ターボ・コンソール用 フォント集
(フォントかしら♪好きさ、大好きさ♪)

その他、便利なツール、PDD、ビープ音、読み物などを満載！
(なお、内容は一部変更されることがあります。ご了承下さい)

編集長祝一平からの御挨拶「どーもどーも。そろそろ桑田君も投げ始めている頃でしょうが、調子が悪いからと言って非難するのは可哀相です。だって昔から言うじゃないですか、♪クワッタ負〜けたと〜騒ぐじゃないぜ〜♪ なんてねーナハハハ」

満開製作所 電脳倶楽部 編集部
〒171 東京都豊島区要町1-19-3 いさみビル4F
TEL.(03)554-9282/FAX.(03)554-3856

販売方法は通信販売のみです。お申し込みの方法は左記の住所へ現金書留で
定期購読 6ヶ月分 6,000円(消費税込・郵送料サービス)
●5月18日以降に受け付けた分は，原則としてVol.25から発送します。新たに購読を希望される方は，「新規」と御明記下さい。
●郵便振替を御利用の場合は口座番号「東京5-362847 満開製作所」でお願いいたします。
製品の性格上，返品には応じられませんが，お申し出があれば定期購読を解約し残金をお返しします。
(ご注意：バックナンバーの受け付けは，定期購読の方に限らせていただきます)

クリエイト特典

- ●全商品完全保証書付(メーカー保証)
- ●全国無料配達(一部離島の方は有料になります)
- ●配達日の指定OK(日曜・祭日にかかわらずお客様のご都合にあわせて配達します)
- ●どんな商品の組合せも自由自在(ご予算、用途に応じ自由自在にシステムアップできます)
- ●中古パソコン高額下取り(今お使いのパソコンをわずかな差額でグレードアップ)
- ●お支払い方法自由(低金利の均等払い、ボーナス一括払いもご利用ください)

営業時間(定休日▶渋谷店:日曜・祭日/横浜店:水曜)
AM10:00〜PM7:00

当社はX68000の販売認定店です。どんなことでも安心してご相談ください。

X68000 ボーナスセール開催中!!

X68000 NEW PRO II

- ●CZ-653C(本体)……………………¥285,000
- ●CZ-603D(カラーディスプレイ)……………………¥ 84,800
- ●お好きなゲームソフト1本……………………¥ 7,800
- ■定価合計……………………¥377,600

クリエイト特価

均等払い	¥ 7,680×48回	¥ 9,890×36回	¥14,370×24回
ボーナス	なし	なし	なし

X68000 NEW EXPERT II

- ●CZ-603C(本体)……………………¥338,000
- ●CZ-613D(カラーディスプレイテレビ)……………………¥ 99,800
- ●CZ-8NJ2……………………¥ 23,800
- ●お好きなゲームソフト1本……………………¥ 9,800
- ■定価合計……………………¥506,600

クリエイト特価

均等払い	¥ 9,970×48回	¥12,840×36回	¥18,660×24回
ボーナス	なし	なし	なし

X68000 EXPERT II HD

- ●CZ-613C(本体)……………………¥448,000
- ●CZ-604D(カラーディスプレイ)……………………¥ 94,800
- ●お好きなゲームソフト1本……………………¥ 9,800
- ■定価合計……………………¥552,600

クリエイト特価

均等払い	¥ 5,920×48回	¥ 7,400×36回	¥12,100×24回
ボーナス	¥30,000× 8回	¥40,000× 6回	¥50,000× 4回

X68000 SUPER HD

- ●CZ-623C-TN(本体・キーボード・マウス)……………………¥498,000
- ●CZ-613D-TN(カラーディスプレイ)……………………¥135,000
- ●CZ-6BP1……………………¥ 79,800
- ■定価合計……………………¥712,800

クリエイト特価

均等払い	¥ 7,320×48回	¥10,100×36回	¥13,450×24回
ボーナス	¥42,000× 8回	¥50,000× 6回	¥80,000× 4回

※本広告に掲載の全商品の価格について消費税は含まれておりません。

X68000シリーズ用 周辺機器・ソフト オール超特価!!

型番	品名	定価	ソフト名	品名	定価
CZ-6VT1	カラーイメージユニット	¥ 69,800	MUSIC PRO	MIDI版	¥ 28,800
CZ-8NS1	カラーイメージスキャナ	¥188,000	MUSIC PRO-68K	マウスを使った楽譜ワープロ	¥ 18,800
CZ-6BE1A	1MB増設RAMボード	¥ 38,000	SOUND PRO-68K	サウンドエディタ	¥ 15,800
CZ-6BE2	2MB増設RAMボード	¥ 79,800	Sampling PRO-68K	AD PCMサンプリングエディタ	¥ 17,800
CZ-6BE4	4MB増設RAMボード	¥138,000	Musicstudio PRO-68K V.1.1	MIDIマルチレコーディングソフト	¥ 28,800
CZ-8NM3	マウス・トラックボール	¥ 9,800	NEW Print Shop PRO-68K	ポップアートツール	¥ 19,800
BF-68PRO	高性能CRTフィルター	¥ 19,800	Communication PRO-68K	高機能通信ソフト	¥ 19,800
CZ-6BP1	数値演算プロセッサ・ボード	¥ 79,800	OS-9/X68000	マルチタスクオペレーティングシステム	¥ 29,800
CZ-8NT1	トラックボール	¥ 13,800	PRO-68K	サイバーノート	¥ 19,800
CZ-6BM1	MIDIボード	¥ 26,800	PRO-68K	ステーショナリー	¥ 14,800
AN-S100	アンプ内蔵スピーカーシステム	¥ 36,600	DATA PRO-68K	コマンド型リレーショナルデータベース	¥ 58,000
CZ-8NJ2	アナログスティック	¥ 23,800	CARD PRO-68K	カード型リレーショナルデータベース	¥ 29,800
CZ-603D	ドットピッチ0.31mm14型高解像度	¥ 84,800	Ccompiler PRO-68K	ソフト開発セット	¥ 39,800
CZ-6TU	パソコンチューナ	¥ 33,100	Human 68K Ver2.0	開発ツールセット	¥ 9,800

▲上記以外ビジネスソフト、最新ゲームソフト豊富に在庫あります。※送料はご注文の際お問合せください。 ●超特価販売中!

★この表以外の組合せ、お支払い方法もご自由にできます。
★X1シリーズ用、X68000シリーズ用各社ハードディスク/プリンタ等の周辺機器を大特価にて販売しております。
電話にてお問合せください。

オール15%〜20%OFF

パソコン専門ショップ

総合お問合せ先☎03-486-6541(代)

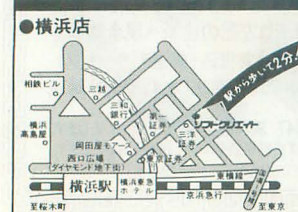
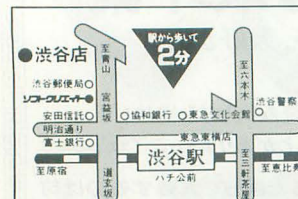
ソフトクリエイト 渋谷/横浜

●渋谷店

●横浜店

●渋谷店☎03-486-6541(代)
〒150:東京都渋谷区渋谷1-12-7 三和渋谷ビル
振込銀行:三井銀行 渋谷宮益坂支店㊬No.5000340

●横浜店☎045-314-4777(代)
〒221:横浜市神奈川区鶴屋町2-12-8 第1建設ビル
振込銀行:三和銀行 横浜駅前支店㊬No.310852

「ゲームは作るのが面白い。」という方を大募集しております。

1. **即戦力プログラマー**（正社員・契約社員・外部スタッフ）
各機種（PC−98・PCエンジン）によるゲーム開発アセンブラが使いまわせる。その上Cが使えるとすごい。

2. **ゲームデザイナー**（契約社員・外部スタッフ・アルバイト）
企画持込み、または、制作アシスタント。

3. **発展途上プログラマー**（契約社員・外部スタッフ・アルバイト）
各機種（PC−98・PCエンジン）の開発アシスタント主にアルバイト。要はやる気と性格です。基本的には将来の工画堂メインプログラマー養成の部署です。メインプログラマーにはりついてノウハウを吸収して下さい。学生歓迎。

4. **移植プログラマー**（契約社員・外部スタッフ）
PC−98で開発されたゲームを他機種（PC−88・MSX2・X68000・FM TOWNS・PCエンジン）へ移植。

5. **グラフィックデザイナー**（契約社員・外部スタッフ）
原画（キャラクターデザイン）または、コンピュータによる自宅作業の可能な方。

※問い合わせ先
〒162　東京都新宿区市谷台町11　TEL03−353−7724
(株)工画堂スタジオ内　ソフトウエア開発部　担当　立岡

※〆切　平成2年7月末日
（右記を原則としますが、どうしてもという方は〆切以降でもご連絡下さい。）

Q：今までどんなゲームをつくりましたか？

学生の特権は 無限大ネットワーク。

データ収集/X-MODEM

レポート提出日が迫ってるけど、作業がはかどらない。助けを求めると、いろんな人が資料を送ってくれた。X−MODEMだと、グラフもそのまま送ってもらえるので大助かり。ボクも“おかえし”しなくては!

夏でもスキー/SIG

スキー大好き!!のボクは1年中スキーと離れられない。SIGにはそんな仲間がいっぱい。シーズン中のスキーツアーはもちろん、オフにもあれこれ情報交換しながら、熱い思いを語り合っている。

家族と交信/電子メール

ふるさとの弟は高校生。ボクと同じ大学をめざしてるので、次々と電子メールで情報をきいてくる。返事と一緒に近況をメールしてたら、おふくろや親父もメールの仲間入り。家族でワイワイ交換日記。

おもしろSHOP探訪/BBS

大学に入学して住んだ初めてのこの街。おもしろSHOPや、心のなごむ公園、とっておきのデートスポット……。BBS(電子掲示板)でたずねると、誰かが教えてくれたから、ボクもすっかりこの街の人。

同郷会/OLT(チャット機能)

BBSやSIGで呼びかけて同郷の人を募り、待ち合わせてOLT(オンライントーク)。なつかしい故郷の話題もさることながら、思わぬ人との出会いがいっぱいあって、充実!同郷のよしみっていいなあ。

自炊の味方/データベース

はじめての自炊。安くて簡単で、しかも栄養のあるものを!そこで大活躍するのが、思いきって買った電子レンジ。データベースの“電子レンジ教室”で、レパートリーはぐんぐん広がる。

僕は一人暮らしの大学3回生。J&P HOT LINEでキャンパスライフプラスαをエンジョイ中。

アクセスポイント新設のお知らせ

東京・大阪・名古屋2400bpsをはじめ、富山・大津・津・堺・熊本1200bpsアクセスポイントを新設しました。全国どこからでも、ますます利用しやすくなったJ&P HOT LINE。あなたも右記のスタータキットで仲間入りしませんか?

J&P HOT LINEは全国90ヵ所のアクセスポイント。2万5千人の仲間が、あなたの仲間になってくれます。

パソコン/ワープロ通信ネットワークサービス

J&P HOT LINE

アクセスポイントは全国に90ヵ所。日本全国を網羅する、本格的な通信ネットワークです。

ご入会はスタータキットで買ったその日からアクセスできます。

■申込先
〒556 大阪市浪速区日本橋西1-6-5 上新電機株式会社
J&P HOT LINE事務局宛 TEL.(06)632-2521

■利用料金について
入会金/3,000円(スタータキット購入の代金から充当されます)
接続料/3分あたり20円(アクセスポイントまでの電話代は含みません)
※消費税3%が加算されます。

スタータキット申込書

お名前	
ご住所	〒
お電話番号	

お申込品 スタータキット(ソフトなし)
3.000+90(消費税3%)=¥3.090

スタータキットのお求めはJ&P各店でどうぞ。

店名	住所	電話
谷 店	東京都渋谷区道玄坂2丁目28番4号	☎(03) 496-4141
田 店	東京都町田市森野1丁目39番16号	☎(0427)23-1313
王子店	東京都八王子市旭町1番1号八王子そごう7F	☎(0426)26-4141
川 店	東京都立川市幸町4−39−1	☎(0425)36-4141
厚木店	厚木市中町3-4-3	☎(0462)25-1548
山 店	富山市桜町2-1-10	☎(0764)32-3133
沢 店	金沢市入江2-63	☎(0762)91-1130
地 店	金沢市寺地2-3	☎(0762)47-2524
須 店	名古屋市中区大須4丁目2-48	☎(052)262-1141
テクノランド	大阪市浪速区日本橋5丁目6番7号	☎(06) 634-1211
メディアランド	大阪市浪速区日本橋5丁目8番26号	☎(06) 634-1511
コスモランド	大阪市浪速区難波中2丁目1番17号	☎(06) 634-3111
U.S.LAND	大阪市浪速区日本橋4丁目9番15号	☎(06) 634-1411
ビジネスランド	大阪市北区梅田1-1-3大阪駅前第3ビルB2	☎(06) 348-1881
梅田店	大阪市北区小松原町1-10	☎(06) 362-1141
高槻店	高槻市高槻町11番16号	☎(0726)85-1212
くずは店	枚方市楠葉花園町15番2号	☎(0720)56-8181
千里中央店	豊中市新千里東町1-3 SENCHU PAL2番街4F	☎(06) 834-4141
摂津富田店	高槻市大畑町24−10	☎(0726)93-7521
寝屋川店	寝屋川市緑町4−20	☎(0720)34-1166
藤井寺店	藤井寺市岡2丁目1番33号	☎(0729)38-2111
岸和田店	岸和田市土生町2451−3	☎(0724)37-1021
さんのみや1ばん館	神戸市中央区八幡通3-2-16	☎(078)231-2111
西宮店	兵庫県西宮市河原町5−11	☎(0798)71-1171
姫路店	姫路市東延末1丁目1番住友生命姫路南ビル1F	☎(0792)22-1221
京都寺町店	京都市下京区寺町通仏光寺下ル恵美須之町549	☎(075)341-3571
京都近鉄店	京都市下京区烏丸通七条下ル東塩小路町702	☎(075)341-5769
和歌山店	和歌山市元寺町4丁目4番地	☎(0734)28-1441
奈良1ばん館	奈良市三条町478−1	☎(0742)27-1111
郡山インター店	大和郡山市横田693−1	☎(07435)9-2221
熊本店	熊本市手取本町4−12	☎(096)359-7800

ADVANCED

先駆の"Z"アビリティがパソコンクリエイターを魅了する。

TURBO

X1 パソコンテレビ turbo Z III

パーソナルコンピュータ＋キーボード＋マウス	CZ-888C-BK	標準価格	169,800円(税別)
14型カラーディスプレイテレビ	CZ-860D-BK	標準価格	92,200円(税別)
チルトスタンド	CZ-6ST1-B	標準価格	5,800円(税別)

クリエイティブマインドを刺激するAV機能　テレビ、ビデオ、ビデオディスクなどの映像を最大4,096色のリアルな画像で瞬時にグラフィック画面に取り込めるカラー画像デジタイズ機能を標準装備。4段階の量子化取り込み、42通りのモザイク取り込みなど多彩なトリック取り込み処理もサポート。さらにクロマキー合成、インターレーススーパーインポーズ、4,096色対応デジタルテロッパ機能、ステレオFM音源…先駆のAV機能がアートワークの領域をさらに拡げます。

AV指向の高水準ベーシックZ-BASIC搭載　多色グラフィック、カラー画像処理、ステレオFM音源、バンクメモリ対応など、ターボZシリーズが本来もつクリエイティブな機能をフルサポート。また豊富な画面モードで多色を駆使するときに便利なグラフィック用関数(HSV, RGB, HALF, CDOWN, CUP)も装備。さらにFM音源制御用ステートメントとしてX68000と命令コンパチの拡張MMLの採用によりスムーズな8音同時演奏を実現しています。

●メインメモリ128Kバイト標準装備、Z-BASICで最大576Kバイトまでサポート●1Mバイトの5インチフロッピーディスクドライブ2基搭載●JIS第1/第2水準準拠漢字、「システム・ユーザー辞書」を標準装備した高度な日本語処理機能●ニューデザインのマウス標準装備●X1ターボシリーズの豊富なソフト資産が活用できるコンパチブル設計●プリンタ、RS-232Cなど豊富なインターフェイスを装備●ドットピッチ0.39mmのハイコントラストブラウン管、15kHz/24kHzのデュアルスキャン方式採用14型カラーディスプレイテレビ(別売)。

資料請求券
X1 turbo ZIII
oh!X
6係

シャープ株式会社　●お問い合わせは…シャープ(株)電子機器事業本部システム機器営業部 〒545 大阪市阿倍野区長池町22番22号☎(06)621-1221(大代表)
電子機器事業本部液晶映像システム事業部第2商品企画部 〒162東京都新宿区市谷八幡町8番地☎(03)260-1161(大代表)

本広告に掲載しております商品および役務の価格には消費税は含まれておりませんので、ご購入の際、消費税額をお支払い下さい。

T4910217906780　雑誌02179-6